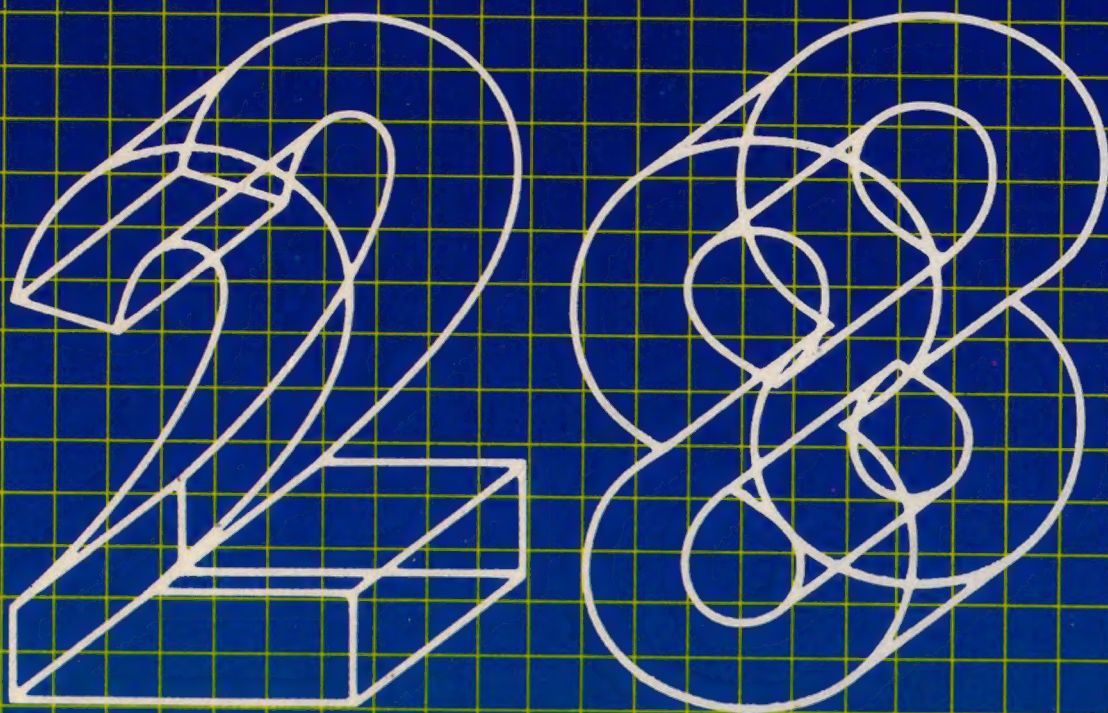


BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



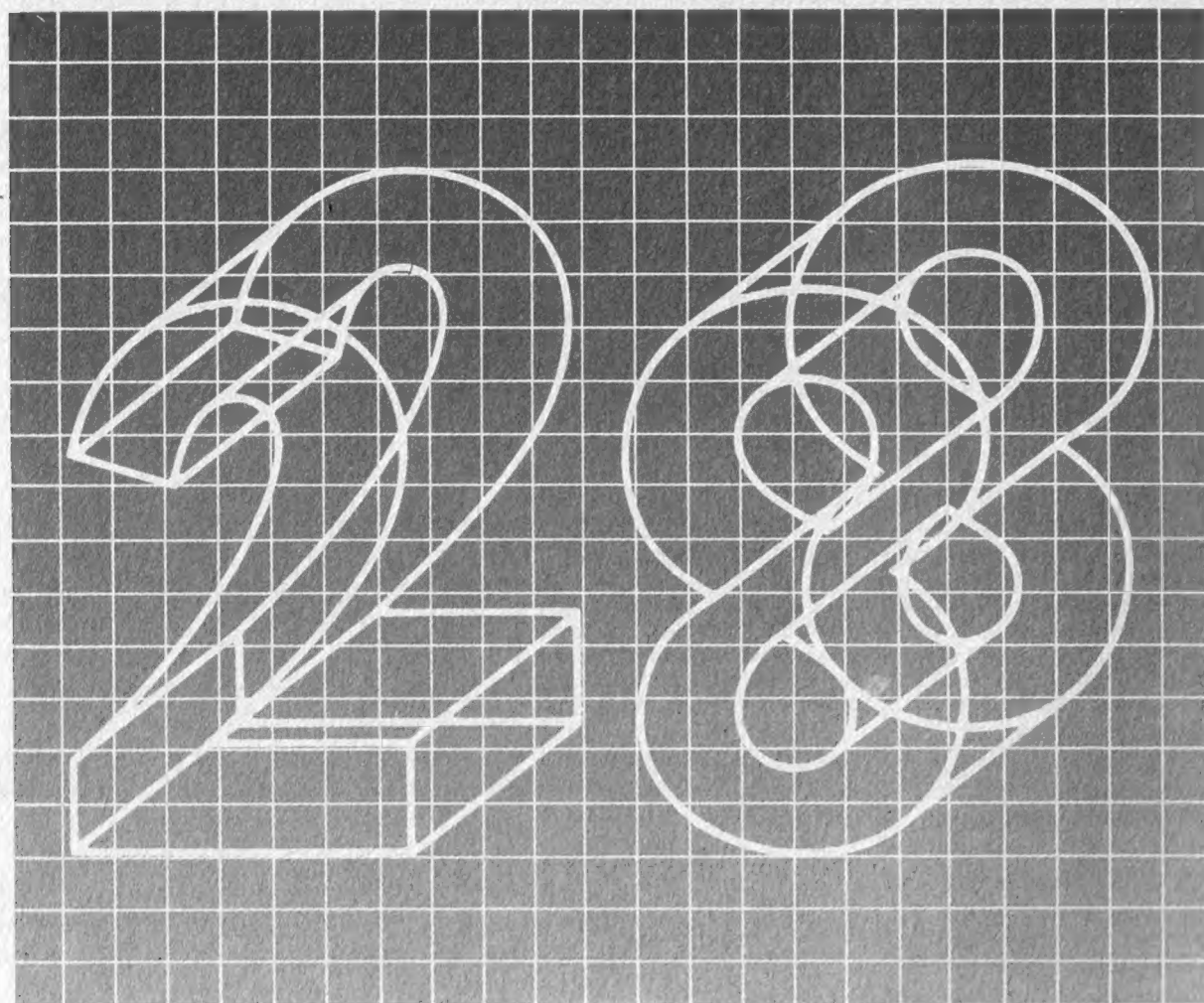
PROGRAMAS EDUCATIVOS,
DE GRAFICOS Y DE UTILIDAD

EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



M

EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 28. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: Jesús Bocho, Licenciado en Informática; Alejandro Marcos, Licenciado en Química. «El Taller de Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-150-9

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

GRAFICAS EMA. (Madrid)

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Agosto, 1987

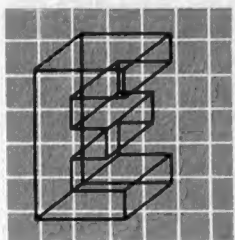
P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	14
■ TRUCOS Y RUTINAS BASICAS	24
■ EL TALLER DEL HARDWARE	33
■ APRENDER CON EL ORDENADOR	40
■ PEQUEÑA HISTORIA DE LA INFORMATICA	51
■ TEMAS MONOGRAFICOS DE VANGUARDIA	55
■ TERMINOLOGIA	59
■ VOCABULARIO DE INFORMATICA	60

EXPERIENCIA Y PRACTICAS EN LOGO

■ Variables para PONCURSOR



A sabemos que la orden PONCURSOR necesita de dos datos para que se pueda ejecutar. Uno que corresponde a la columna y otro que corresponde a la fila.

En un principio vimos que estos datos no podían ser los valores correspondientes a variables, sino que teníamos que indicar la fila y la columna por medio de valores constantes.

Por ejemplo, si definimos el siguiente procedimiento:

```
? PARA EJEMPLO
> HAZ "C 10
> HAZ "F 15
> PONCURSOR [:C :F]
> ES "HOLA
> FIN
```

y lo ejecutamos:

```
? EJEMPLO
```

obtendríamos el siguiente mensaje de error:

```
PONCURSOR NO ACEPTA 10 COMO
DATO EN EJEMPLO
```

Al intentar ejecutar la orden PONCURSOR se encuentra con una variable como dato, que aunque tiene un valor determinado no lo entiende.

El Logo no permite, sin embargo, poder utilizar variables con esta orden. Para ello lo que tenemos que hacer es transformar en una lista los datos que tenemos en las variables.

En el ejemplo anterior, con sólo realizar un cambio, podemos conseguirlo:

```
? PARA EJEMPLO
> HAZ "C 10
```

```
> HAZ "F 15
> PONCURSOR LISTA :C :F
> ES "HOLA
> FIN
```

Añadimos la orden LISTA delante de las variables para que PONCURSOR se pueda ejecutar.

Si el valor que tiene asignado una variable es superior al de filas o columnas que tiene la pantalla, obtendremos el mismo mensaje de error que si se le ha asignado directamente:

```
? PONCURSOR [100 10]
PONCURSOR NO ADMITE 100 COMO
DATO
? HAZ "C 100
? HAZ "F 10
? PONCURSOR LISTA :C :F
PONCURSOR NO ADMITE 100 COMO
DATO
```

Aquí tienes ejemplos para ver cómo actúa la orden PONCURSOR cuando utiliza datos variables:

El siguiente procedimiento rellena con la letra A una zona de la pantalla comprendida entre la columna 0 y la 10 y la fila 0 y la 10:

```
? PARA EJEMPLO1
> HAZ "C AZAR 11
> HAZ "F AZAR 11
> PONCURSOR LISTA :C :F
> ES [A]
> EJEMPLO1
> FIN
```

Si lo ejecutamos:

```
? EJEMPLO1
```

en la pantalla aparecerá lo siguiente:

Un objeto Logo contiene un número determinado de elementos.



Fig. 1.

A las variables C (columna) y F (fila) le asignamos un valor aleatorio entre 0 y 10. A continuación, el cursor se coloca en la posición que indican los valores que tienen las variables y se escribe una A. Una vez escrita, el procedimiento se vuelve a ejecutar y se escribe otra A en la misma o en otra posición.

Este procedimiento al ser recursivo no pararía jamás por sí solo.

El siguiente procedimiento escribe en todas las posiciones de la línea 10 una X:

```
? PARA EJEMPLO2
> PONCURSOR LISTA :C :F
> ES "X
> SI :C = 38 [ALTO] [HAZ "C :C + 1]
> EJEMPLO2
> FIN
```

En este caso tenemos que inicializar las variables antes de ejecutar el procedimiento:

```
? HAZ "C 0
? HAZ "F 10
```

Si ahora lo ejecutamos:

```
? EJEMPLO2
```

Obtenemos una pantalla como ésta:

Inicializamos la variable de las columnas a cero y la de las filas a diez.

La primera posición sería, pues, la [0 10] y en ella es escribir la primera X.

A continuación preguntamos si la variable de las columnas tiene el valor 38. De ser así detenemos el proceso, ya que no se podría escribir más. Si no lo es, incrementamos en

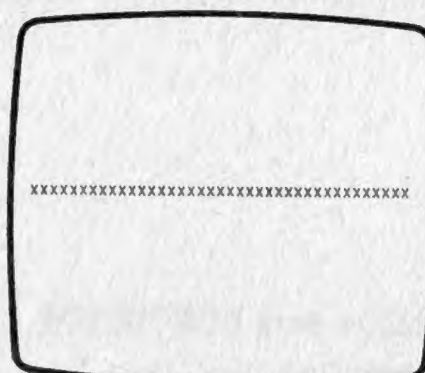


Fig. 2.

uno el valor de la variable de las columnas y volvemos a ejecutar el procedimiento.

Por último, el siguiente procedimiento rellena toda la pantalla con la letra O:

```
? PARA EJEMPLO3
> PONCURSOR LISTA :C :F
> ES "O
> HAZ "F :F + 1
> SI :F = 24 [HAZ "F 0 HAZ "C :C + 1]
> SI :C = 38 [ALTO]
> EJEMPLO3
> FIN
```

Antes de ejecutar este procedimiento tenemos que inicializar también las variables:

```
? HAZ "C 0
? HAZ "F 0
```

Si ahora lo ejecutamos:

```
? EJEMPLO3
```

La pantalla tendrá el siguiente aspecto:

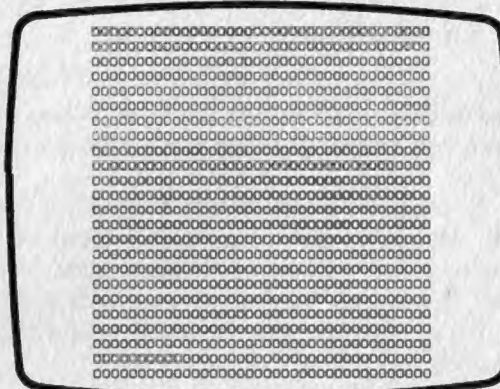


Fig. 3.

Si el objeto es una palabra, los elementos son caracteres.

En este ejemplo vamos aumentando la variable de las filas desde cero hasta 24 por cada vez que estamos en una columna. En el momento que hemos alcanzado la fila 24, significa que hemos llegado a la última fila de la pantalla, por lo que volvemos a situarnos en la fila 0 y avanzamos una columna. Este proceso se repetirá hasta que la variable de las columnas tenga el valor 38.

■ Operaciones con objetos

— MIEMBRO? objeto1 objeto2

Devuelve CIERTO si objeto1 es un elemento de objeto2, si no, devuelve FALSO.

? ESCRIBE MIEMBRO? "A "CASA
CIERTO

? ESCRIBE MIEMBRO? "R "SACO
FALSO

? ESCRIBE MIEMBRO? [LUNES] [[LUNES] MARTES MIERCOLES]
CIERTO

? ESCRIBE MIEMBRO? [LUNES] [LUNES MARTES MIERCOLES]
FALSO

La lista [LUNES] no es un elemento de la lista [LUNES MARTES MIERCOLES].

? ESCRIBE MIEMBRO? "LUNES [LUNES MARTES MIERCOLES]
CIERTO

La palabra LUNES si es un elemento de la lista [LUNES MARTES MIERCOLES].

NOTA: En el LOGO del SPECTRUM la sintaxis de esta primitiva es:

PERTENECE? objeto [lista]

— CUENTA objeto

Devuelve el número de elementos del objeto especificado.

? ESCRIBE CUENTA "TELEVISOR
9

La palabra TELEVISOR tiene nueve caracteres.

? ESCRIBE CUENTA "A
1

? ESCRIBE CUENTA "
0

La palabra vacía no contiene ningún elemento.

? ESCRIBE CUENTA [ESTO ES ASI]
3

La lista [ESTO ES ASI] contiene tres elementos (palabras).

? ESCRIBE CUENTA [[HOLA] ADIOS [HASTA LA VISTA]]
3

La lista especificada contiene tres elementos: dos listas y una palabra.

? ES CUENTA []
0

La lista vacía no contiene ningún elemento.

El número que devuelve la primitiva CUENTA podemos almacenarlo en una variable.

— ELEMENTO n objeto

Devuelve el n-ésimo elemento del objeto especificado.

? ESCRIBE ELEMENTO 2 "CASA
A

? ESCRIBE ELEMENTO 5 "UTIL
L

? ESCRIBE 1 [LUNES MARTES MIERCOLES]
LUNES

Si queremos obtener el último elemento de un objeto determinado, lo podemos hacer así:

? PARA ULTI :A
> HAZ "E CUENTA :A
> ESCRIBE ELEMENTO :E :A
> FIN

Para obtener, por ejemplo, el último elemento de la palabra ESTUDIO ejecutamos el procedimiento que acabamos de definir indicando la palabra en cuestión.

? ULTI "ESTUDIO
0

Si el objeto es una lista, los elementos son palabras y/o listas.

EXPERIENCIA Y PRACTICAS EN LOGO

En la pantalla aparece la letra O, que es la última de la palabra ESTUDIO.

Otro ejercicio en el que utilizamos las primitivas CUENTA y ELEMENTO puede ser:

Definir un procedimiento que devuelve los elementos de un objeto en orden contrario.

```
? PARA P :A
> HAZ "E CUENTA :A
> HAZ "C O
> REPITE :E [HAZ "S :E - :C ESCRIBE
ELEMENTO :S :A HAZ "C :C + 1]
> FIN
```

Al procedimiento P le pasamos al ejecutarlo el objeto que deseamos escribir al revés.

Probemos con una palabra:

```
? P "SIMPATICO
```

O
C
I
T
A
P
M
I
S

Probemos con una lista:

```
? P [NO LLEVO PARAGUAS Y ESTA
LLOVIENDO A CANTAROS]
```

CANTAROS
A
LLOVIENDO
ESTA
Y
PARAGUAS
LLENO
NO

Usando un procedimiento recursivo podemos conseguir lo mismo que con P.

```
? PARA P1 :A :E
> SI :E = 0 [ALTO]
> ESCRIBE ELEMENTO :E :A
> P1 :A :E - 1
> FIN
```

El procedimiento P1 tiene dos entradas:

A es el objeto

E es el número de elementos del objeto

Al ejecutar P1 debemos indicar dos datos.

Para no tener que escribir tanto cada vez que utilicemos P1, definimos otro procedimiento.

```
? PARA "REVES :A
> HAZ "E CUENTA :A
> P1 :A :E
> FIN
```

Probemos con una palabra:

```
? REVES "SEMAFORO
```

O
R
O
F
A
M
E
S

Probemos con una lista:

```
? REVES [AL QUE MADRUGA DIOS LE
AYUDA]
```

AYUDA
LE
DIOS
MADRUGA
QUE
AL

■ Cuadro resumen

MIEMBRO? objeto1 objeto2

Devuelve CIERTO si objeto1 es un elemento de objeto2, si no, devuelve FALSO.

En el LOGO del SPECTRUM la sintaxis es:

PERTENECE? objeto [lista]

— CUENTA objeto

Si quieres utilizar variables para la orden PONCURSOR, no te olvides que tienes que utilizar la orden LISTA.

Devuelve el número de elementos del objeto especificado.

— ELEMENTO n objeto

Devuelve el n-ésimo elemento de objeto.

■ Ejercicios

1. Forma con una palabra, por ejemplo ORDENADOR, un triángulo como el de la figura:

ORDENADOR
ORDENADO
ORDENAD
ORDENA
ORDEN
ORDE
ORD
OR
O

2. ¿Cómo podemos saber el número de caracteres que contiene un objeto cualquiera?

Intenta resolver con los procedimientos necesarios este pequeño problema.

NOTA: Los objetos pueden ser palabras o listas que contengan sólo palabras.

No tendremos en cuenta los espacios en blanco.

3. Intenta realizar el siguiente dibujo:

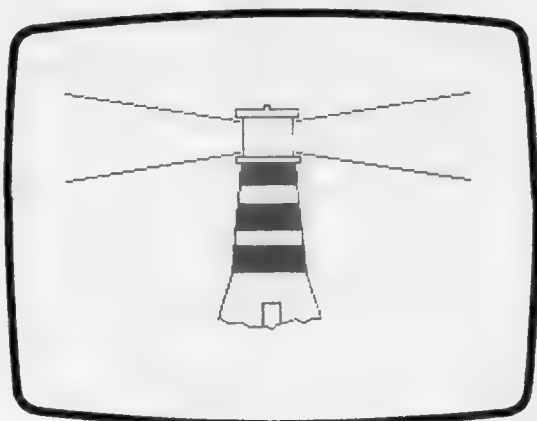


Fig. 4.

4. Define los procedimientos necesarios para realizar el siguiente proceso:

1. Partiendo de las posiciones en que están los siguientes asteriscos:

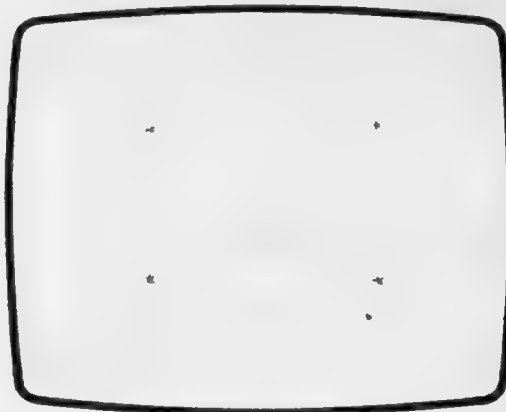


Fig. 5.

Escribir asteriscos hasta que confluyan en la posición central.

2. Hacer parpadear el asterisco central cinco veces.

3. Sustituir, partiendo de la posición central, los asteriscos por puntos.

4. Hacer parpadear los puntos finales cinco veces.

5. ¿Son correctas las siguientes órdenes?:

- ? HAZ "C 10
- ? HAZ "F 20
- ? PONCURSOR [:C :F]
- ? ES "LOGO
- ? PARA A
 - > HAZ "C AZAR 0
 - > HAZ "F AZAR 24
 - > PONCURSOR LISTA :C :F
 - > ES "Y
 - > A
 - > FIN
- ? HAZ "C 5
- ? HAZ "F 10
- ? PONCURSOR LISTA [:C :F]
- ? PARA A1
 - > HAZ "A "TITULO DEL LIBRO
 - > HAZ "C CUENTA :A
 - > FIN
- ? PARA A2
 - > HAZ "A []

Averigua el n-ésimo elemento de cualquier objeto con la primitiva ELEMENTO.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> ESCRIBE ELEMENTO 0 :A
> FIN

- ? PARA A3
> HAZ "A [ADIOS S]
> SI MIEMBRO? "S :A [ESCRIBE
"VERDADERO] [ES "FALSO]
> FIN
```

■ Solución a los ejercicios

1:

El procedimiento TRI almacena en la variable E la longitud de la palabra A y llama al procedimiento de dos entradas FORMA.

```
? PARA TRI :A
> HAZ "E CUENTA :A
> FORMA :A :E
> FIN
```

El procedimiento recursivo FORMA tiene dos entradas:

A es la palabra
E es la longitud de la palabra

Se encarga de escribir en la pantalla la palabra contenida en A, la palabra contenida en A menos el último carácter, la palabra contenida en A menos dos caracteres..., y el primer carácter de la palabra contenida en A.

```
? PARA FORMA :A :E
> HAZ "C :E - 1 HAZ "P "
> REPITE :E [HAZ "S :E - :C
HAZ "P PALABRA :P (ELEMENTO :S
:A) HAZ "C :C - 1]
> ESCRIBE :P
> SI :E ≠ 1 [ALTO] [FORMA :A :E - 1]
> FIN
```

Formemos un triángulo con la palabra CORRECAMINOS

```
? TRI "CORRECAMINOS
```

En la palabra aparece:

CORRECAMINOS
CORRECAMINO
CORRECAMIN
CORRECAMI
CORRECAM
CORRECA
CORREC

```
CORRE  
CORR  
COR  
CO  
C
```

2:

Con el procedimiento P5 leemos una lista del teclado, guardamos en la variable E el número de elementos de la lista, inicializamos la variable contadora a cero y llamamos al procedimiento CONTAR.

```
? PARA P5
> HAZ "A LL
> HAZ "E CUENTA :A
> HAZ "D 0
> CONTAR :A :E :D
> FIN
```

El procedimiento CONTAR acumula en la variable D el número de caracteres de cada una de las palabras que componen la lista introducida.

```
? PARA CONTAR :A :E :D
> HAZ "EL ELEMENTO :E :A
> HAZ "D :D + CUENTA :EL
> SI :E = A [ESCRIBE :D ALTO] [CONTAR :E - 1 :D]
> FIN
```

Se repite su ejecución hasta que la variable E (número de elementos de la lista) vale 1.

Ejemplo:

```
? P5
EL PROCEDIMIENTO P5 FUNCIONA BIEN
29
```

La lista introducida contiene 29 caracteres.

3:

Para dibujar el faro vamos a definir varios procedimientos que dibujan diferentes partes de éste.

En primer lugar, definimos el que va a dibujar la torre:

```
? PARA TORRE
> SL
> PONPOS [-25 -40]
> BL
> GD 15 AV 30
```

La palabra vacía no contiene ningún elemento.

- > GI 10 AV 70
- > PONRUMBO 90
- > AV 30 GD 85
- > AV 70 GI 10
- > AV 30
- > PONRUMBO 270
- > AV 5 GI 30
- > AV 5 GD 55
- > AV 5 GI 50
- > AV 6 GD 20
- > AV 5 GI 45
- > AV 4 GD 60
- > AV 5 GI 10
- > AV 8 GD 25
- > AV 6 GI 20
- > AV 8 GD 45
- > AV 5
- > FIN

La puerta la dibuja PUERTA:

? PARA PUERTA

- > SL
- > PONPOS [0 -45]
- > PONRUMBO 0
- > BL
- > AV 15 GD 90
- > AV 10 GD 90
- > AV 13
- > FIN

Para dibujar la franjas, definimos uno que, en primer lugar, traza las líneas de las franjas y a continuación rellena las diferentes zonas:

? PARA FRANJAS

- > PONRUMBO 90
- > SL
- > PONPOS [-16 -10]
- > BL
- > AV 40
- > SL
- > PONPOS [-16 6]
- > BL
- > AV 40
- > SL
- > PONPOS [-14 16]
- > BL
- > AV 37
- > SL
- > PONPOS [-13 32]
- > BL

- > AV 33
- > SL
- > PONPOS [-11 45]
- > BL
- > AV 30
- > SL
- > PONPOS [-5 -2]
- > PONRUMBO 0
- > REPITE 3 [BL RELLENA SL AV 24]
- > FIN

La parte superior del faro la dibuja el procedimiento que llamamos SUPE:

? PARA SUPE

- > SL
- > PONPOS [-14 59]
- > PONRUMBO 0
- > BL
- > REPITE 2 [AV 4 GD 90 AV 36 GD 90]
- > AV 4 GD 90
- > AV 4 GI 90
- > AV 25 GD 90
- > AV 28 GD 90
- > AV 25
- > RE 25 GI 90
- > AV 3 GI 90
- > AV 4 GI 90
- > AV 16 GD 90
- > AV 3 GI 90
- > AV 3 GI 90
- > AV 3 GD 90
- > AV 16 GI 90
- > AV 4 GI 90 AV 3
- > FIN

Por último, definimos el que dibuja los rayos que emite el faro:

? PARA RAYO

- > SL
- > PONPOS [-12 85]
- > PONRUMBO 280
- > BL
- > AV 100
- > SL
- > PONPOS [-12 65]
- > PONRUMBO 260
- > BL
- > AV 100
- > SL
- > PONPOS [20 85]
- > PONRUMBO 80

Ahora podemos utilizar variables con la orden PONCURSOR.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> BL
> AV 100
> SL
> PONPOS [20 65]
> PONRUMBO 100
> BL
> AV 100
> FIN
```

Una vez que tenemos definidos los procedimientos que dibujan las diferentes partes del faro; definimos uno para agruparlos e inicializar el proceso:

```
? PARA FARO
> BL
> OT
> TORRE
> PUERTA
> FRANJAS
> SUPE
> RAYO
> FIN
```

Para ejecutarlo, bastará con introducir:

```
? FARO
```

4:

En primer lugar, definimos el procedimiento A, que es el que sitúa los asteriscos en la posición de partida:

```
? PARA A
> BT
> PONCURSOR [9 0]
> ES ""
> PONCURSOR [29 0]
> ES ""
> PONCURSOR [9 20]
> ES ""
> PONCURSOR [29 20]
> ES ""
> FIN
```

Con el procedimiento B conseguimos escribir un asterisco desde cada una de las posiciones iniciales hasta el centro:

```
? PARA B
> HAZ "A 9
> HAZ "B 29
> HAZ "C 0
> HAZ "D 20
> REPITE 10 [HAZ "A :A + 1
```

```
HAZ "C :C + 1
PONCURSOR LISTA :A :C
ES "" HAZ "B :B - 1
PONCURSOR LISTA :B :C
ES "" HAZ "D :D - 1
PONCURSOR LISTA :A :D
ES "" PONCURSOR LISTA :B :D
ES ""]
```

```
> REPITE 10 [PONCURSOR [19 10]
ES "" ESPERA 5 PONCURSOR [19 10]
ES ". ESPERA 5]
> FIN
```

El final de este procedimiento realiza el parpadeo del asterisco central.

El siguiente procedimiento es el que, partiendo de la posición central, sustituye los asteriscos por puntos:

```
? PARA C
> HAZ "A 19
> HAZ "B 19
> HAZ "C 10
> HAZ "D 10
> REPITE 10 [HAZ "A :A - 1
HAZ "C :C - 1
PONCURSOR LISTA :A :C ES "
HAZ "B :B + 1
PONCURSOR LISTA :B :C
ES ". HAZ "D :D + 1
PONCURSOR LISTA :A :D
ES ". PONCURSOR LISTA :B :D
ES ".]
> REPITE 5 [D E]
> FIN
```

Al final de este procedimiento repetimos cinco veces la llamada a los procedimientos C y D, que son los que realizan el parpadeo de los puntos finales:

```
? PARA D
> PONCURSOR LISTA :A :C
> ES ""
> ESPERA 1
> PONCURSOR LISTA :B :C
> ES ""
> ESPERA 1
> PONCURSOR LISTA :A :D
> ES ""
> ESPERA 1
> PONCURSOR LISTA :B :D
> ES ""
```

Podemos escribir aleatoriamente en cualquier parte de la pantalla.

```

> ESPERA 1
> FIN

? PARA E
> PONCURSOR LISTA :A :C
> ES ".
> ESPERA 1
> PONCURSOR LISTA :B :C
> ES ".
> ESPERA 1
> PONCURSOR LISTA :A :D
> ES ".
> ESPERA 1
> PONCURSOR LISTA :B :D
> ES ".
> ESPERA 1
> FIN

```

Para realizar todo el proceso seguido, definimos un último procedimiento para englobarlos e inicializar el proceso:

```

? PARA TODO
> BT
> A
> B
> C
> FIN

```

Para ejecutarlo, introducimos:

```

? TODO

```

5:

```

— ? HAZ "C 10
  ? HAZ "F 20
  ? PONCURSOR [:C :F]
  ? ES "LOGO

INCORRECTA.

```

No se puede indicar a la orden PONCURSOR dos datos contenidos en variables directamente.

```

— PARA A
  > HAZ "C AZAR 0
  > HAZ "F AZAR 24
  > PONCURSOR LISTA :C :F
  > ES "Y
  > A
  > FIN

CORRECTA.

```

La letra Y se escribirá aleatoriamente en cualquier fila de la columna cero.

```

— ? HAZ "C 5
  ? HAZ "F 10
  ? PONCURSOR LISTA [:C :F]

INCORRECTA.

```

No se pueden poner los corchetes.

```

— ? PARA A1
  > HAZ "A "TITULO DEL LIBRO
  > HAZ "C CUENTA :A
  > FIN

INCORRECTO.

```

Intentamos asignar a la variable A una frase sin encerrarla entre corchetes.

EL LOGO muestra el mensaje de error:
NO SE COMO HACER PARA DEL EN A1

```

— ? PARA A2
  > HAZ "A []
  > ESCRIBE ELEMENTO 0 :A
  > FIN

INCORRECTO.

```

El número *n* que acompaña a la primitiva ELEMENTO debe ser un entero distinto de cero.

El LOGO muestra el mensaje:

ELEMENTO NO ACEPTA O COMO DATO.

```

— ? PARA A3
  > HAZ "A [ADIOS]
  > SI MIEMBRO? "S :A [ESCRIBE "VERDADERO] [ES "FALSO]
  > FIN

CORRECTO.

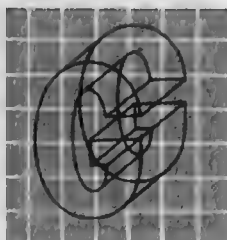
```

Si la lista [ADIOS] contiene al elemento S, en la pantalla aparece la palabra VERDADERO, si no, aparece la palabra FALSO.

Cuando ejecutemos el procedimiento A3 en la pantalla aparecerá FALSO, ya que S no es un elemento de la lista en cuestión; el único elemento de la lista [ADIOS] es la palabra ADIOS.

No excedas el valor de las variables que utilizas para posicionar el cursor.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



CONTINUANDO con el juego de ejemplo que hemos podido realizar con todo lo que hemos visto hasta ahora, vamos a ver a continuación la versión del PAC-MAN en el AMSTRAD.

Esta versión es más parecida a la del IBM que la del SPECTRUM. Fijese el lector que se han definido dos monstruos distintos con dos posiciones cada uno. Esto ha sido posible gracias a los caracteres definidos de que disponemos.

```

1000 REM *****
1010 REM *
1020 REM *   P A C   M A N   *
1030 REM *
1040 REM *****
1050 REM
1060 REM *****
1070 REM *
1080 REM *   A U T O R   *
1090 REM *
1100 REM *
1110 REM * Fco. Morales Guerrero *
1120 REM *
1130 REM *****
1140 REM
1150 REM *****
1160 REM *
1170 REM * (c) Ed. Siglo Cultural *
1180 REM * (c) 1987 *
1190 REM *
1200 REM *****
1210 REM
1220 REM *****
1230 REM * INICIALIZACION DEL PROGRAMA *
1240 REM *****
1250 REM
1260 DIM N(9),M$(10),P(10)
1270 DIM S(25,41),M$(2,3),I(2,8)
1280 LET M$(1,0)=""
1290 LET M$(1,1)=""
1300 LET M$(1,3)=""
1310 LET M$(2,0)=""
1320 LET M$(2,1)=""
1330 LET M$(2,3)=""
1340 LET PA=1
1350 MODE 1
1360 PEN 3
1370 CLS
1380 PRINT TAB(15); "COME = COCOS"
1390 PRINT TAB(14); "=====

```

```

1400 PEN
1410 LOCATE 10,14:PRINT "CARGANDO DATAS"
1420 PEN 2
1430 LOCATE 12,12:PRINT "ESPERA UN MOMENTO."
1440 REM
1450 REM *****
1460 REM * DEFINICION DE LOS PERSONAJES *
1470 REM *****
1480 REM
1490 SYMBOL 255,60,126,255,255,255,126,60 : REM PAC-MAN CERRADO
1500 SYMBOL 254,60,126,255,15,15,255,126,60 : REM PAC-MAN IZQUIERDA
1510 SYMBOL 253,60,126,255,240,240,255,126,60 : REM PAC-MAN DERECHA
1520 SYMBOL 252,36,102,231,231,255,255,126,60 : REM PAC-MAN ARRIBA
1530 SYMBOL 251,60,126,255,255,231,231,102,36 : REM PAC-MAN ABAJO
1540 SYMBOL 250,60,126,219,219,126,60,36,36 : REM FANTASMA-1 POSICION A
1550 SYMBOL 249,60,126,219,219,126,60,36,36 : REM FANTASMA-1 POSICION B
1560 SYMBOL 248,126,255,189,219,126,36,66,129 : REM FANTASMA-2 POSICION A
1570 SYMBOL 247,126,255,189,219,126,36,36,36 : REM FANTASMA-2 POSICION B
1580 REM
1590 REM *****
1600 REM * LECTURA DE LOS CARACTERES *
1610 REM *****
1620 REM
1630 FOR I=0 TO 9
1640   READ N(I)
1650 NEXT I
1660 REM
1670 REM *****
1680 REM * LECTURA DE LOS RECORDS *
1690 REM *****
1700 REM
1710 FOR I=1 TO 10
1720   READ N$(I),P(I)
1730 NEXT I
1740 REM
1750 REM *****
1760 REM * LECTURA DE LAS DIRECCIONES *
1770 REM *****
1780 REM
1790 FOR I=1 TO 2
1800   FOR J=1 TO 8
1810     READ I(I,J)
1820   NEXT J
1830 NEXT I
1840 REM
1850 REM *****
1860 REM * PROGRAMA PRINCIPAL *
1870 REM *****
1880 REM
1890 CLS
1900 GOSUB 5590:REM PRESENTACION
1910 GOSUB 4730:REM TABLA DE RECORDS
1920 GOSUB 5340:REM INSTRUCCIONES
1930 GOSUB 5490:REM PULSA UNA TECLA
1940 GOSUB 4200:REM DIBUJA LA PANTALLA
1950 PEN 2
1960 GOSUB 2020:REM LEE EL TECLADO
1970 GOSUB 2170:REM MUEVE EL COME-COME
1980 GOSUB 2530:REM ENEMIGO NUMERO 1
1990 GOSUB 2760:REM ENEMIGO NUMERO 2
2000 IF PT=0 THEN GOSUB 3730:LET PA=PA+1:GOTO 1940
2010 GOTO 1960
2020 REM
2030 REM *****
2040 REM * CONTROL DEL TECLADO *
2050 REM *****

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

2070 LET IS=INKEY$: IF IS="" THEN LET IS=Z$
2080 LET Z$=IS
2090 LET II=INSTR("OoPpQqAa", IS)
2100 IF II=0 THEN RETURN
2110 LET X1=X0: LET Y1=Y0
2120 LET X0=X0+I(1, II)+I(1, II)*(S(Y0, X0+I(1, II))=0)
2130 LET Y0=Y0+I(2, II)+I(2, II)*(S(Y0+I(2, II), X0)=0)
2140 IF X0=41 THEN LET X0=1
2150 IF X0=0 THEN LET X0=40
2160 RETURN
2170 REM
2180 REM *****
2190 REM * MOVIMIENTO DEL COME-COME *
2200 REM *****
2210 REM
2220 IF BB>0 THEN LET BB=BB+1: IF BB=60-(10*PA) THEN LET BB=0
2230 IF X1=X0 AND Y1=Y0 THEN LET Z$="Z": RETURN
2240 IF S(Y0, X0)=-1 THEN LET PT=PT-1: LET S(Y0, X0)=2: LET PU=PU+10: LOCATE 13, 2: PRINT USING "#####"; PT
2250 IF S(Y0, X0)=1 THEN LET PT=PT-1: LET S(Y0, X0)=2: LET PU=PU+50: LOCATE 13, 2: PRINT USING "#####"; PT: LET PT=1
2260 IF PU>HI THEN LOCATE 32, 2: PRINT USING "#####"; PU: LET HI=PU
2270 LOCATE X1, Y1: PRINT
2280 LET S(Y1, X1)=2
2290 IF SW=1 THEN LOCATE X0, Y0: PRINT CHR$(255): LET SW=0: LET S(Y0, X0)=99: RETURN
2300 ON II GOSUB 2330, 2330, 2350, 2350, 2350, 2430, 2430, 2480, 2480
2310 LET SW=1: LET S(Y0, X0)=99
2320 RETURN
2330 REM
2340 REM *** MOVIMIENTO A LA IZQUIERDA ***
2350 REM
2360 LOCATE X0, Y0: PRINT CHR$(254)
2370 RETURN
2380 REM
2390 REM *** MOVIMIENTO A LA DERECHA ***
2400 REM
2410 LOCATE X0, Y0: PRINT CHR$(253)
2420 RETURN
2430 REM
2440 REM *** MOVIMIENTO A ARRIBA ***
2450 REM
2460 LOCATE X0, Y0: PRINT CHR$(252)
2470 RETURN
2480 REM
2490 REM *** MOVIMIENTO A ABAJO ***
2500 REM
2510 LOCATE X0, Y0: PRINT CHR$(251)
2520 RETURN
2530 REM
2540 REM *****
2550 REM * MOVIMIENTO DEL FANTASMA-1 *
2560 REM *****
2570 REM
2580 REM *** CALCULO DE LA DIRECCION ***
2590 REM
2600 LET SO=0: LET MM=MM+1
2610 IF BB>0 AND SD1=1 THEN LET SD1=0: RETURN
2620 LET SD1=1
2630 LET X3=X2: LET Y3=Y2
2640 LET SX=SGN(X1-X2)
2650 LET SY=SGN(Y1-Y2)
2660 LET X2=X2+SX+SX*(S(Y2, X2+SX)=0)
2670 LET Y2=Y2+SY+SY*(S(Y2+SY, X2)=0)
2680 REM
2690 REM *** MOVIMIENTO ***

```

```

2700 REM
2710 LOCATE X3,Y3:PRINT M$(1,M1+1)
2720 LET M1=S(Y2,X2)
2730 IF M1=99 THEN GOTO 3010
2740 IF SE1=1 THEN LET SE1=0:LOCATE X2,Y2:PRINT CHR$(250):RETURN
2750 LET SE1=1:LOCATE X2,Y2:PRINT CHR$(249):RETURN
2760 REM
2770 REM *****
2780 REM * MOVIMIENTO DEL FANTASMA-2 *
2790 REM *****
2800 REM
2810 REM *** CALCULO DE LA DIRECCION ***
2820 REM
2830 LET SQ=0
2840 IF BB=0 AND LET SD2=1 THEN LET SD2=0:RETURN
2850 LET SD2=1
2860 IF X3=X4 AND Y3=Y4 THEN LET MM=9
2870 IF MM<15 THEN LOCATE X4,Y4:PRINT CHR$(248):RETURN
2880 LET X5=X4:LET Y5=Y4
2890 LET SX=SGN(X1-X4)
2900 LET SY=SGN(Y1-Y4)
2910 LET X4=X4+SX+SX*(S(Y4,X4+SX)=0)
2920 LET Y4=Y4+SY+SY*(S(Y4+SY,X4)=0)
2930 REM
2940 REM *** MOVIMIENTO ***
2950 REM
2960 LOCATE X5,Y5:PRINT M$(2,M2+1)
2970 LET M2=S(Y4,X4)
2980 IF M2=99 THEN GOTO 3010
2990 IF SE2=1 THEN LET SE2=0:LOCATE X4,Y4:PRINT CHR$(247):RETURN
3000 LET SE2=1:LOCATE X4,Y4:PRINT CHR$(248):RETURN
3010 REM
3020 REM *****
3030 REM * PERDIDA DE UNA VIDA *
3040 REM *****
3050 REM
3060 LOCATE X0,Y0:PRINT " "
3070 FOR I=1 TO 10
3080   LOCATE X0,Y0:PRINT CHR$(255)
3090   FOR K=1 TO 100
3100     NEXT K
3110   LOCATE X0,Y0:PRINT " "
3120   FOR K=1 TO 100
3130     NEXT K
3140 NEXT I
3150 LOCATE Y0,X0:PRINT " "
3160 LOCATE Y2,X2:PRINT " "
3170 LOCATE Y4,X4:PRINT " "
3180 LET NV=NV-1
3190 IF NV=0 THEN GOTO 3440
3200 LET Y0=18
3210 LET X0=20
3220 LET Y1=Y0
3230 LET X1=X0
3240 LET Z$="Z"
3250 LET BB=0
3260 LET X2=19
3270 LET Y2=11
3280 LET X3=X2
3290 LET Y3=Y2
3300 LET X4=21
3310 LET Y4=11
3320 LET X5=X4
3330 LET Y5=Y4
3340 LET M1=S(Y2,X2)
3350 LET M2=S(Y4,X4)

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

3360 LOCATE 10,24:PRINT
3370 FOR I=1 TO NV-1
3380   LOCATE 9+I,24:PRINT CHR$(253)
3390 NEXT I
3400 LOCATE X0,Y0:PRINT CHR$(243)
3410 GOSUB 3810
3420 RETURN
3430 RETURN
3440 REM
3450 REM *****
3460 REM * GAME OVER *
3470 REM *****
3480 REM
3490 FOR I=1 TO 5
3500   FOR J=1 TO 3
3510     PEN J
3520     LOCATE 18,13:PRINT " GAME "
3530     LOCATE 18,14:PRINT " OVER "
3540     FOR K=1 TO 140:NEXT K
3550     NEXT J
3560     LOCATE 18,13:PRINT "
3570     LOCATE 18,14:PRINT "
3580     FOR K=1 TO 140:NEXT K
3590 NEXT I
3600 GOTO 1910
3610 REM
3620 REM *****
3630 REM * ESTAS LISTO *
3640 REM *****
3650 REM
3660 LOCATE 18,13:PRINT (ESTAS"
3670 LOCATE 18,14:PRINT LISTO?"
3680 FOR J=1 TO 1000
3690 NEXT J
3700 LOCATE 18,13:PRINT "
3710 LOCATE 18,14:PRINT "
3720 RETURN
3730 REM
3740 REM *****
3750 REM * TERMINO DE UNA PANTALLA *
3760 REM *****
3770 REM
3780 FOR I=1 TO 10
3790   BORDER 12
3800   FOR J=1 TO 100
3810     NEXT J
3820   BORDER 0
3830   FOR J=1 TO 100
3840     NEXT J
3850 NEXT I
3860 CLS
3870 PEN 3
3880 PRINT "MUY BIEN"
3890 PRINT
3900 PRINT TAB(10);"TERMINASTE LA PANTALLA"
3910 FOR I=2 TO 40
3920   LOCATE I,10
3930   PRINT " "
3940 NEXT I
3950 FOR I=1 TO 10
3960   LOCATE I,10
3970   PRINT CHR$(253)
3980   FOR J=1 TO 100
3990     NEXT J
4000   LOCATE I,10
4010   PRINT "

```

```

4020   FOR J=1 TO 100
4030   NEXT J
4040 NEXT I
4050 LOCATE 1,10
4060 PRINT CHR$(253)
4070 FOR I=2 TO 40
4080   LOCATE I,10:PRINT " ";CHR$(253)
4090   FOR J=1 TO 100:NEXT J
4100   LOCATE I,10:PRINT " ";CHR$(255)
4110   FOR J=1 TO 100:NEXT J
4120 NEXT I
4130 PRINT
4140 PRINT " HAS CONSEGUIDO "
4150 PRINT
4160 PRINT PA*800;"PUNTOS DE BONUS"
4170 LET PU=PU+PA*800
4180 GOSUB 5490
4190 RETURN
4200 REM
4210 REM *****
4220 REM * DIBUJO DE LA PANTALLA *
4230 REM *****
4240 REM
4250 CLS
4260 RESTORE 6390
4270 FOR I=1 TO 25
4280   IF I=25 THEN LOCATE 1,25
4290   READ A$
4300   FOR J=1 TO LEN(A$)
4310     LET S(I,J)=0
4320     LET B$=MID$(A$,J,1)
4330     IF B$="." OR B$="*" THEN PEN 2:PRINT B$;LET S(I,J)=(B$=".")-(B$="*")
PEN 3:GOTO 4390
4340     IF B$>"9" THEN GOTO 4360
4350     PRINT CHR$(N(VAL(B$)));GOTO 4390
4360     IF B$="A" THEN PRINT CHR$(155);
4370     IF B$="B" THEN PRINT CHR$(158);
4380     IF B$="C" THEN PRINT CHR$(140);
4390   NEXT J
4400 NEXT I
4410 PEN 3
4420 LOCATE 4,2:PRINT USING "PUNTOS = #####";PU
4430 LOCATE 27,2:PRINT USING "HI = #####";HI
4440 LOCATE 2,24:PRINT "VIDAS = ";
4450 LOCATE 28,24:PRINT USING "PANTALLA = #";PA;
4460 FOR I=1 TO NV-1
4470   LOCATE 9+I,24:PRINT CHR$(253)
4480 NEXT I
4490 LET Y0=16
4500 LET X0=20
4510 LET Y1=Y0
4520 LET X1=X0
4530 LET Z$="Z"
4540 LET S(13,41)=2
4550 LET S(13,0)=2
4560 LET BB=0
4570 LET MM=0
4580 LET M2=-1
4590 LET PT=287
4600 LET X2=19
4610 LET Y2=11
4620 LET X3=X2
4630 LET Y3=Y2
4640 LET X4=21
4650 LET Y4=11
4660 LET X5=X4

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

4680 LET M1=-1
4690 LOCATE X0,Y0
4700 PRINT CHR$(255)
4710 GOSUB 3610
4720 RETURN
4730 REM
4740 REM *****
4750 REM * TABLA DE RECORDS *
4760 REM *****
4770 REM
4780 CLS
4790 PEN 3
4800 PRINT TAB(19);"TABLA DE RECORDS"
4810 PRINT TAB(12);"===== "
4820 PEN 1
4830 PRINT TAB(6);CHR$(150);STRING$(28,154);CHR$(156)
4840 PRINT TAB(6);CHR$(149);SPC(28);CHR$(149)
4850 PRINT TAB(6);CHR$(151);STRING$(28,154);CHR$(151)
4860 FOR I=1 TO 9
4870   PRINT TAB(6);CHR$(149);SPC(28);CHR$(149)
4880 NEXT I
4890 PRINT TAB(6);CHR$(147);STRING$(28,154);CHR$(153)
4900 NP=0
4910 FOR I=1 TO 10
4920   IF PU>P(I) THEN LET NP=1
4930 NEXT I
4940 IF NP=1 THEN LET P(10)=PU:LET N$(10)=CHR$(255)
4950 FOR I=1 TO 10
4960   FOR J=1 TO I
4970     IF P(I)>P(J) THEN LET A=N$(I):LET A=P(I):LET N$(I)=N$(J):LET P(I)=P(
J):LET N$(J)=A:LET P(J)=A
4980   NEXT J
4990 NEXT I
5000 IF NP=0 THEN GOTO 5050
5010 FOR I=1 TO 10
5020   IF N$(I)=CHR$(255) THEN LET N$(I)=STRING$(15,"-"):LET NP=I
5030 NEXT I
5040 PEN 2
5050 LOCATE 7,4:PRINT N$(1);TAB(23);"... ";USING "#####";P(1)
5060 FOR I=2 TO 10
5070   LOCATE 7,4+I
5080   PRINT N$(I);TAB(23);"... ";USING "#####";P(I)
5090 NEXT I
5100 IF NP=0 THEN GOTO 5290
5110 PEN 3
5120 LOCATE 15,16:PRINT "FELICIDADES."
5130 PRINT
5140 PRINT TAB(3);"TU PUNTUACION ESTA ENTRE LAS MEJORES"
5150 PRINT
5160 PRINT TAB(12);"ESCRIBE TU NOMBRE."
5170 LET N$(NP)="" :LET X=7:LET LO=0:LET D$=""
5180 IF NP=1 THEN LET Y=4 ELSE LET Y=4+NP
5190 LOCATE X,Y
5200 PRINT "...";CHR$(29);
5210 LET A$=INKEY$:IF A$="" THEN GOTO 5210
5220 IF A$=CHR$(8) AND LO>0 THEN LET X=X-1:LET LO=LO-1:LET D$=LEFT$(D$,LO):PRINT
CHR$(29);" _";CHR$(29);CHR$(29);
5230 IF A$="" THEN GOTO 5260
5240 IF A$=CHR$(13) THEN GOTO 5270
5250 IF A$<"0" OR A$>"z" THEN GOTO 5210
5260 PRINT A$;" _";CHR$(29):LET LO=LO+1:LET D$=D$+A$:IF LO>15 THEN GOTO 5210
5270 PRINT SPC(16-LO)
5280 LET N$(NP)=D$
5290 LET H1=P(1)
5300 LOCATE 15,16:PRINT SPACE$(240)

```

```

5310 LET PU=0:LET NVE=3
5320 IF NP<>0 THEN LET NP=0:GOSUB 5590:GOTO 4730
5330 RETURN
5340 REM
5350 REM *****
5360 REM * INSTRUCCIONES *
5370 REM *****
5380 REM
5390 PEN 3
5400 LOCATE 4,17
5410 PRINT "COMETE TODOS LOS PUNTOS QUE PUEDAS"
5420 PRINT
5430 PRINT "      ^"
5440 PRINT "      Q      USA ESTAS TECLAS"
5450 PRINT "< O P >"
5460 PRINT "      A      PARA MOVER EL QUESITO."
5470 PRINT "      v"
5480 RETURN
5490 REM
5500 REM *****
5510 REM * PULSA UNA TECLA *
5520 REM *****
5530 REM
5540 LOCATE 13,25
5550 PRINT "PULSA UNA TECLA"
5560 LET A$=INKEY$
5570 IF A$="" THEN GOTO 5560
5580 RETURN
5590 REM
5600 REM *****
5610 REM * PRESENTACION *
5620 REM *****
5630 REM
5640 CLS
5650 PEN 3
5660 RESTORE 6890
5670 FOR I=2 TO 16
5680   READ A$
5690   FOR J=1 TO LEN(A$)
5700     IF MID$(A$,J,1)="0" THEN GOTO 5730
5710     LOCATE 7+J,I+1
5720     PRINT CHR$(253)
5730   NEXT J
5740 NEXT I
5750 LOCATE 6,1:PRINT "(c) Ed. Siglo Cultural, 1987"
5760 PEN 2
5770 FOR I=1 TO 14
5780   LOCATE I,20:PRINT "."
5790   FOR J=1 TO 100:NEXT J
5800 NEXT I
5810 PEN 3
5820 LOCATE 17,20:PRINT "= 10 PUNTOS"
5830 PEN 2
5840 FOR I=1 TO 13
5850   LOCATE I,22:PRINT " *"
5860   FOR J=1 TO 100:NEXT J
5870 NEXT I
5880 PEN 3
5890 LOCATE 17,22:PRINT "= 50 PUNTOS"
5900 FOR J=1 TO 1000:NEXT J
5910 FOR I=1 TO 15
5920   LOCATE I,20:PRINT CHR$(253)
5930   FOR J=1 TO 100:NEXT J
5940   LOCATE I,20:PRINT CHR$(255)
5950   FOR J=1 TO 100:NEXT J
5960   LOCATE I,20:PRINT " "

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

5970 NEXT I
5980 LOCATE 15,20:PRINT CHR$(253)
5990 LOCATE 17,20:PRINT "- TU MISMO "
6000 FOR I=1 TO 15
6010   LOCATE I,22:PRINT CHR$(250)
6020   FOR J=1 TO 100:NEXT J
6030   LOCATE I,22:PRINT CHR$(249)
6040   FOR J=1 TO 100:NEXT J
6050   LOCATE I,22:PRINT " "
6060 NEXT I
6070 LOCATE 15,22:PRINT CHR$(250)
6080 LOCATE 17,22:PRINT "- TU ENEMIGO"
6090 FOR I=1 TO 1000
6100 NEXT I
6110 RETURN
6120 REM
6130 REM *****
6140 REM * DATAS CON LOS CARACTERES *
6150 REM *****
6160 REM
6170 DATA 32,150,156,147,153,154,148,159,151,157
6180 REM
6190 REM *****
6200 REM * DATAS CON LOS RECORDS *
6210 REM *****
6220 REM
6230 DATA "ANTONIO",34980
6240 DATA "EL COCO",31720
6250 DATA "EL AUTOR",24030
6260 DATA "INDIANA",23300
6270 DATA "JOSELITO",11210
6280 DATA "SUSYN",11120
6290 DATA "RAMIRO",8430
6300 DATA "DONALD",7540
6310 DATA "SHE",3400
6320 DATA "HE",3210
6330 REM
6340 REM *****
6350 REM * DATAS CON LAS DIRECCIONES *
6360 REM *****
6370 REM
6380 DATA -1,-1,1,1,0,0,0,0,0,0,0,0,-1,-1,1,1
6390 REM
6400 REM *****
6410 REM * DATAS CON LA DEFINICION DE LA PANTALLA *
6420 REM *****
6430 REM
6440 DATA "0015555555555555555520000000155555555555200"
6450 DATA "0060000000000000000000000000000000000000000000"
6460 DATA "15A55555555555555555B5A55555555555555A52"
6470 DATA "6*.....35555554.....*6"
6480 DATA "6.15552.1555552.....1555552.15552.6"
6490 DATA "6.60006.600000352.1552.154000006.60006.6"
6500 DATA "6.35554.355555554.6006.355555554.35554.6"
6510 DATA "6.....6006.....6"
6520 DATA "8555552.155552.1554003552.155552.1555559"
6530 DATA "8155526.601554.3555555554.355206.8155526"
6540 DATA "6355546.606.....606.6355546"
6550 DATA "3555554.354.152.10000002.152.354.3555554"
6560 DATA ".....606.60000006.606....."
6570 DATA "1555552.152.606.60000006.606.152.1555552"
6580 DATA "6...34.354.354.35555554.354.354.34...6"
6590 DATA "6.12.....12.6"
6600 DATA "6.63552.155555555555555555555555552.15546.6"
6610 DATA "6.35526.6000155555555555555520006.61554.6"
6620 DATA "6...34.35554.....35554.34...6"

```

```

6630 DATA "6.12.....15B55555B52.....12.6"
6640 DATA "6.34.15555552.354.....354.15555552.34.6"
6650 DATA "8*...35555208.....1552.....60155554...*8"
6660 DATA "8555555555A5755555A55A5555BA5A555555559"
6670 DATA "800000000000800000000000006000000000008"
6680 DATA "35555555555400000000000003555555555554"
6690 REM
6700 REM *****
6710 REM * DATAS CON LA PALABRA PAC-MAN *
6720 REM *****
6730 REM
6740 DATA "1111110000110000011110"
6750 DATA "0110011001111000110011"
6760 DATA "0110011011001101100000"
6770 DATA "0111100011001101100000"
6780 DATA "0110000011111101100000"
6790 DATA "0110000011001100110011"
6800 DATA "1111000011001100011110"
6810 DATA ""
6820 DATA "11000110000110001100011"
6830 DATA "11101110001111001110011"
6840 DATA "11111110011001101111011"
6850 DATA "11111110011001101101111"
6860 DATA "11010110011111101100111"
6870 DATA "11000110011001101100011"
6880 DATA "11000110011001101100011"

```

TRUCOS Y RUTINAS BASICAS



continuación vamos a ver un programa realizado en el AMSTRAD para definir ventanas en la pantalla. Con este programa podremos definir hasta diez pantallas distintas y ver cómo tendría que ser el programa que nos permitiese dibujarlas en la pantalla. Dicho programa puede ser visualizado en pantalla o impreso por impresora.

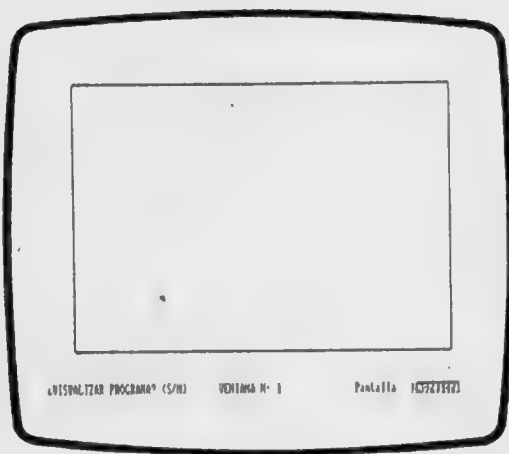


Fig. 1

Lo único que hay que hacer para definir dichas ventanas es mover la flecha con las teclas del cursor y pulsar ENTER cuando nos encontremos en la posición que nosotros queremos que sea la esquina superior derecha. Una vez hecho esto, nos vamos a la esquina inferior derecha y volvemos a pulsar la tecla ENTER. En este momento se visualizará la ventana y el ordenador almacenará los valores necesarios.

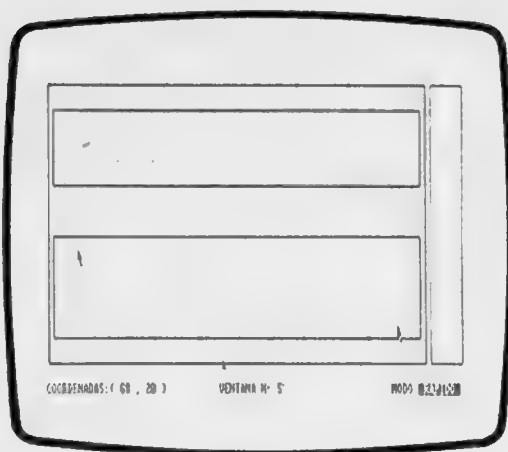


Fig. 2

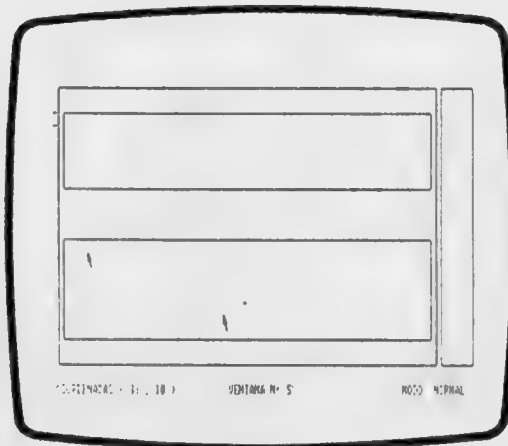


Fig. 3.

Cuando hayamos terminado de definir ventanas, podemos hacer que los listados que nos permiten crearlas aparezcan por pantalla o por impresora.

```

1000 REM *****
1010 REM *
1020 REM *      W I N D O W S      *
1030 REM *
1040 REM *****
1050 REM
1060 REM *****
1070 REM *
1080 REM * Programa realizado por: *
1090 REM *
1100 REM * Carlos A. Maria Morin *
1110 REM *
1120 REM *****
1130 REM
1140 REM *****
1150 REM *
1160 REM * (c) Ediciones Siglo Cultural *
1170 REM * (c) 1987
1180 REM *
1190 REM *****
1200 REM
1210 REM
1220 REM *****
1230 REM ***** PRESENTACION E INICIALIZACION *****
1240 REM *****
1250 REM
1260 MODE 0
1270 LOCATE 4,13
1280 PRINT "W I N D O W S"
1290 FOR RET=1 TO 3000
1300 NEXT RET
1310 MODE 2
1320 LET XX=0
1330 LET YA=363
1340 LET W=0
1350 LET GIGA=1
1360 ORIGIN 0,0
1370 INK 1,26
1380 INK 0,0
1390 SYMBOL 241,128,192,224,240,248,252,254,216
1400 SYMBOL 242,28,12,14,6,0,0,0,0
1410 SYMBOL 243,48,72,48,120,0,0,0,0
1420 PRINT CHR$(23);CHR$(1);
1430 FOR RET=1 TO 100
1440 NEXT RET
1450 REM
1460 REM *****
1470 REM ***** PROGRAMA PRINCIPAL *****
1480 REM *****
1490 REM
1500 LET W=W+1
1510 IF W>10 THEN LET D$="S":GOTO 1980
1520 GOSUB 2580
1530 LOCATE 67,25
1540 PRINT "MODO:"
1550 LOCATE 1,25
1560 PRINT "COORDENADAS:"
1570 LOCATE 34,25
1580 PRINT "VENTANA N";CHR$(243)
1590 GOSUB 2250
1600 GOSUB 2050
1610 LET POX1=X
1620 LET POY1=Y
1630 GOSUB 2050
1640 LET POX2=X
1650 LET POY2=Y
1660 IF POX1=POX2 THEN GOTO 1630

```

TRUCOS Y RUTINAS BASICAS

```

1870 IF POY1>POY2 THEN GOTO 1530
1880 IF POX1>POX2 THEN GOTO 1530
1890 IF POY1>POY2 THEN GOTO 1530
1900 GOSUB 2480
1910 REM
1920 REM *****
1930 REM ***** DIBUJA VENTANA *****
1940 REM *****
1950 REM
1960 LOCATE POX1,POY1
1970 PRINT CHR$(150);
1980 FOR X=POX1+1 TO POX2-1
1990 PRINT CHR$(154);
2000 NEXT X
2010 PRINT CHR$(156);
2020 FOR Y=POY1+1 TO POY2-1
2030 LOCATE POX1,Y
2040 PRINT CHR$(149);
2050 LOCATE POX2,Y
2060 PRINT CHR$(149);
2070 NEXT Y
2080 LOCATE POX1,POY2
2090 PRINT CHR$(147);
2100 FOR X=POX1+1 TO POX2-1
2110 PRINT CHR$(154);
2120 NEXT X
2130 PRINT CHR$(153);
2140 LOCATE 1,25
2150 PRINT "VISUALIZAR PROGRAMAY (S/N)"
2160 LET DS=UPPER$(INKEY$)
2170 IF DS<>"S" AND DS<>"N" THEN GOTO 1960
2180 IF DS="S" THEN GOSUB 2800
2190 GOTO 1500
2200 REM
2210 REM *****
2220 REM ***** MOVIMIENTO DEL CURSOR *****
2230 REM *****
2240 REM
2250 LET AS=UPPER$(INKEY$)
2260 LET X=INT((X%/8)+1)
2270 LET Y=INT((ABS(Y%-400))/16)
2280 IF INKEY(47)=0 THEN LET GIGA=GIGA+1:LET MARCA=1 ELSE LET MARCA=0
2290 IF GIGA=2 THEN LET GIGA=1
2300 IF (GIGA=1 AND MARCA=1) THEN GOSUB 2580
2310 IF (GIGA=2 AND MARCA=1) THEN GOSUB 2690
2320 IF JOY(0)=36 OR INKEY(8)=0 THEN GOSUB 2260:LET XX=MAX(0,XX)-XAUM:GOSUB 2250
2330 IF JOY(0)=40 OR INKEY(1)=0 THEN GOSUB 2260:LET XX=MIN(640,XX)+XAUM:GOSUB 22
50
2340 IF JOY(0)=33 OR INKEY(0)=0 THEN GOSUB 2260:LET YX=MIN(400,YX)+YAUM:GOSUB 22
50
2350 IF JOY(0)=34 OR INKEY(2)=0 THEN GOSUB 2260:LET YX=MAX(0,YX)-YAUM:GOSUB 2250
2360 IF AS="B" THEN GOTO 1310
2370 IF AS=CHR$(13) OR JOY(0)=48 THEN GOSUB 2250:TAGOFF:RETURN
2380 TAGOFF
2390 LOCATE 12,25
2400 PRINT "X":X;"Y":Y
2410 LOCATE 44,25
2420 PRINT W
2430 TAG
2440 GOTO 2050
2450 GOSUB 2380
2460 TAG
2470 REM
2480 MOVE XX,YX+8
2490 PRINT CHR$(241);
2500 MOVE XX,YX-8
2510 PRINT CHR$(242);

```

```

2330 REM
2340 REM *****
2350 REM ***** LIMITES DE LOS BORDES *****
2360 REM *****
2370 REM
2380 IF X%>838 THEN LET X%=0
2390 IF X%<0 THEN LET X%=832
2400 IF Y%<0 THEN LET Y%=383
2410 IF Y%>398 THEN LET Y%=0
2420 RETURN
2430 REM
2440 REM *****
2450 REM ***** MEMORIZACION DE LAS DIEZ VENTANAS *****
2460 REM *****
2470 REM
2480 LET XPOF1(W)=POX1
2490 LET XPOF2(W)=POX2
2500 LET YPOF1(W)=POY1
2510 LET YPOF2(W)=POY2
2520 RETURN
2530 REM
2540 REM *****
2550 REM ***** MODO NORMAL *****
2560 REM *****
2570 REM
2580 TAGOFF
2590 LET XAUM=8
2600 LET YAUM=16
2610 LOCATE 72,25
2620 PRINT "NORMAL"
2630 RETURN
2640 REM
2650 REM *****
2660 REM ***** MODO RAPIDO *****
2670 REM *****
2680 REM
2690 TAGOFF
2700 LET XAUM=32
2710 LET YAUM=64
2720 LOCATE 72,25
2730 PRINT CHR$(24); " RAPIDO ";CHR$(24)
2740 RETURN
2750 REM
2760 REM *****
2770 REM ***** VISUALIZA EL PROGRAMA EN PANTALLA O IMPRESORA *****
2780 REM *****
2790 REM
2800 LET X=80
2810 LET Y=25
2820 LET T1$="Pantalla"
2830 LET T2$="Impresora"
2840 GOSUB 3320
2850 IF OP=-1 THEN LET CANAL=8 ELSE LET CANAL=0
2860 CLS
2870 IF W=11 THEN LET W=10
2880 PRINT "CON QUE NUMERO DE LINEA QUIERES QUE COMIENCE EL LISTADO?"
2890 PRINT:PRINT
2900 INPUT "NUMERO DE LINEA = ";LIN
2910 CLS
2920 FOR BU=1 TO W
2930   FOR X=1 TO 25
2940     PRINT
2950   NEXT
2960   LOCATE 1,1
2970   IF CANAL=8 THEN PRINT#8:PRINT#8
2980   PRINT#CANAL "VENTANA NUMERO ";BU

```

TRUCOS Y RUTINAS BASICAS

```

2990 PRINT#CANAL:PRINT#CANAL:PRINT#PRINT#PRINT#
3000 PRINT#CANAL,LIN;" LOCATE ";XPOR1(BU);";";YPOR1(BU)
3010 PRINT#CANAL,LIN+10;" PRINT CHR$(150)"
3020 PRINT#CANAL,LIN+20;" FOR X=";XPOR1(BU)+1;" TO ";XPOR2(BU)-1
3030 PRINT#CANAL,LIN+30;" PRINT CHR$(154);"
3040 PRINT#CANAL,LIN+40;" NEXT X"
3050 PRINT#CANAL,LIN+50;" PRINT CHR$(156)"
3060 PRINT#CANAL,LIN+60;" FOR Y=";YPOR1(BU)+1;" TO ";YPOR2(BU)-1
3070 PRINT#CANAL,LIN+70;" LOCATE ";XPOR1(BU);";Y"
3080 PRINT#CANAL,LIN+80;" PRINT CHR$(149)"
3090 PRINT#CANAL,LIN+90;" LOCATE ";XPOR2(BU);";Y"
3100 PRINT#CANAL,LIN+100;" PRINT CHR$(149)"
3110 PRINT#CANAL,LIN+110;" NEXT Y"
3120 PRINT#CANAL,LIN+120;" LOCATE ";XPOR1(BU);";";YPOR2(BU)
3130 PRINT#CANAL,LIN+130;" PRINT CHR$(147);"
3140 PRINT#CANAL,LIN+140;" FOR X=";XPOR1(BU)+1;" TO ";XPOR2(BU)-1
3150 PRINT#CANAL,LIN+150;" PRINT CHR$(154);"
3160 PRINT#CANAL,LIN+160;" NEXT X"
3170 PRINT#CANAL,LIN+170;" PRINT CHR$(153)"
3180 LET LIN=LIN+180
3190 LOCATE 32,25
3200 PRINT "(PULSA UNA TECLA)"
3210 WHILE INKEY$=""
3220 WEND
3230 NEXT BU
3240 CLS
3250 LET W=0
3260 RETURN
3270 REM
3280 REM *****
3290 REM ***** SUBROUTINA DE SELECCION DE OPCIONES *****
3300 REM *****
3310 REM
3320 LET E$=CHR$(24)
3330 LET P$=""
3340 LET OP=1
3350 LOCATE X,Y
3360 PRINT E$+T1$+E$+" "+P$+T2$+P$
3370 LET A$=INKEY$
3380 IF A$="" THEN GOTO 3370
3390 IF A$=CHR$(13) THEN RETURN
3400 IF A$<>" " THEN GOTO 3370
3410 LET G$=E$
3420 LET E$=P$
3430 LET P$=G$
3440 LET OP=OP
3450 GOTO 3350

```

VENTANA NUMERO: 1

```

10 LOCATE 1,3:PRINT CHR$(27);
20 FOR X=2 TO 54
30 PRINT CHR$(154);
40 NEXT
50 PRINT CHR$(156)
60 FOR Y=4 TO 9
70 LOCATE 1,Y:PRINT CHR$(149):LOCATE 55,Y:PRINT CHR$(149)
80 NEXT
90 LOCATE 1,10:PRINT CHR$(147);
100 FOR X=2 TO 54
110 PRINT CHR$(154);
120 NEXT
130 PRINT CHR$(153)

```

[PULSA UNA TECLA]

Fig. 4

A continuación aparece este mismo programa en la versión para IBM.

```

1000 REM *****
1010 REM *
1020 REM *      W I N D O W S      *
1030 REM *
1040 REM *****
1050 REM
1060 REM *****
1070 REM *
1080 REM * Programa realizado por: *
1090 REM *
1100 REM * Carlos A. Maria Morin *
1110 REM *
1120 REM *****
1130 REM
1140 REM *****
1150 REM *
1160 REM * (c) Ediciones Siglo Cultural *
1170 REM * (c) 1987
1180 REM *
1190 REM *****
1200 REM
1210 REM
1220 REM *****
1230 REM ***** PRESENTACION E INICIALIZACION *****
1240 REM *****
1250 REM
1260 OPEN "LPT1:" FOR OUTPUT AS #8
1270 OPEN "SCRN:" FOR OUTPUT AS #1
1280 SCREEN 2
1290 WIDTH 40
1300 CLS
1310 LOCATE 8,15
1320 PRINT "W I N D O W S"
1330 LOCATE 9,14
1340 PRINT "-----"
1350 FOR RET=1 TO 1000
1360 NEXT RET
1370 WIDTH 80
1380 DIM NX(10)
1390 LOCATE 1,1
1400 PRINT CHR$(127)
1410 GET(0,0)= (7,7),NX
1420 CLS
1430 LET X%=1
1440 LET Y%=1
1450 LET W=0
1460 LET N=1
1470 LET GIGA=1
1480 REM
1490 REM *****
1500 REM ***** PROGRAMA PRINCIPAL *****
1510 REM *****
1520 REM
1530 LET W=W+1
1540 IF W=10 THEN LET DS="S" GOTO 2020
1550 GOSUB 2600
1560 LOCATE 25,87
1570 PRINT "MODO:"
1580 LOCATE 21,1
1590 PRINT "COORDENADAS:"

```

TRUCOS Y RUTINAS BASICAS

```

1600 LOCATE 25,34
1610 PRINT "VENTANA No."
1620 GOSUB 2300
1630 GOSUB 2110
1640 LET POX1=X
1650 LET POY1=Y
1660 GOSUB 2110
1670 LET POX2=X
1680 LET POY2=Y
1690 IF POX1=POX2 THEN GOTO 1560
1700 IF POY1=POY2 THEN GOTO 1560
1710 IF POX1>POX2 THEN GOTO 1560
1720 IF POY1>POY2 THEN GOTO 1560
1730 GOSUB 2500
1740 REM
1750 REM *****
1760 REM ***** DIBUJA VENTANA *****
1770 REM *****
1780 REM
1790 LOCATE POY1,POX1
1800 PRINT CHR$(201);
1810 FOR X=POX1+1 TO POX2-1
1820   PRINT CHR$(208);
1830 NEXT X
1840 PRINT CHR$(187);
1850 FOR Y=POY1+1 TO POY2-1
1860   LOCATE Y,POX1
1870   PRINT CHR$(188);
1880   LOCATE Y,POX2
1890   PRINT CHR$(188);
1900 NEXT Y
1910 LOCATE POY2,POX1
1920 PRINT CHR$(200);
1930 FOR X=POX1+1 TO POX2-1
1940   PRINT CHR$(205);
1950 NEXT X
1960 PRINT CHR$(188);
1970 LOCATE 25,1
1980 PRINT "VISUALIZAR PROGRAMA? (S/N)"
1990 LET D$=INKEY$
2000 IF D$="" THEN GOTO 1990
2010 IF D$<>"S" AND D$<>"N" THEN GOTO 1990
2020 IF D$="S" THEN GOSUB 2300
2030 LOCATE 25,1
2040 PRINT "
2050 GOTO 1630
2060 REM
2070 REM *****
2080 REM ***** MOVIMIENTO DEL CURSOR *****
2090 REM *****
2100 REM
2110 LET A$=INKEY$
2120 IF A$="" THEN GOTO 2110
2130 LET X=X%
2140 LET Y=Y%
2150 IF A$=" " THEN LET GIGA=GIGA+1:LET MARCA=1 ELSE LET MARCA=0
2160 IF GIGA>2 THEN LET GIGA=1
2170 IF (GIGA=1 AND MARCA=1) THEN GOSUB 2600
2180 IF (GIGA=2 AND MARCA=1) THEN GOSUB 2700
2190 IF A$="O" THEN GOSUB 2310:LET X=X%-XAUM:GOSUB 2300
2200 IF A$="P" THEN GOSUB 2310:LET X=X%+XAUM:GOSUB 2300
2210 IF A$="A" THEN GOSUB 2310:LET Y=Y%+YAUM:GOSUB 2300
2220 IF A$="Q" THEN GOSUB 2310:LET Y=Y%-YAUM:GOSUB 2300
2230 IF A$="B" THEN RUN
2240 IF A$=CHR$(13) THEN GOSUB 2300:RETURN
2250 LOCATE 25,13

```

```

2260 PRINT "X", "Y"
2270 LOCATE 25,44
2280 PRINT W
2290 GOTO 2110
2300 GOSUB 2400
2310 REM
2320 REM
2330 PUT((X%-1)*8+(Y%-1)*8),N%.XOR
2340 RETURN
2350 REM
2360 REM *****
2370 REM ***** LIMITE DE LOS BORDES *****
2380 REM *****
2390 REM
2400 IF X%>80 THEN LET X%=1
2410 IF X%<1 THEN LET X%=80
2420 IF Y%<1 THEN LET Y%=24
2430 IF Y%>24 THEN LET Y%=1
2440 RETURN
2450 REM
2460 REM *****
2470 REM ***** MEMORIZACION DE LAS DIEZ VENTANAS *****
2480 REM *****
2490 REM
2500 LET XPOR1(W)=POX1
2510 LET XPOR2(W)=POX2
2520 LET YPOR1(W)=POY1
2530 LET YPOR2(W)=POY2
2540 RETURN
2550 REM
2560 REM *****
2570 REM ***** MODO NORMAL *****
2580 REM *****
2590 REM
2600 LET XAUM=1
2610 LET YAUM=1
2620 LOCATE 25,72
2630 PRINT "NORMAL "
2640 RETURN
2650 REM
2660 REM *****
2670 REM ***** MODO RAPIDO *****
2680 REM *****
2690 REM
2700 LET XAUM=5
2710 LET YAUM=5
2720 LOCATE 25,72
2730 PRINT " RAPIDO "
2740 RETURN
2750 REM
2760 REM *****
2770 REM ***** VISUALIZA EL PROGRAMA EN PANTALLA O IMPRESORA *****
2780 REM *****
2790 REM
2800 LET X=13
2810 LET Y=25
2820 LET T$(1)="Pantalla "
2830 LET T$(2)="Impresora"
2840 LOCATE 25,1
2850 PRINT "SALIDA POR: Pantalla"
2860 GOSUB 3370
2870 IF N=2 THEN LET CANAL=8 ELSE LET CANAL=1
2880 CLS
2890 IF W=11 THEN LET W=10
2900 PRINT "CON QUE NUMERO DE LINEA QUIERES QUE COMIENCE EL LISTADO?"
2910 PRINT:PRINT

```

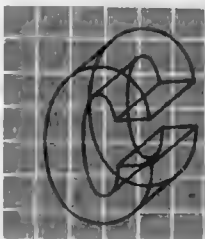
TRUCOS Y RUTINAS BASICAS

```

2920 INPUT "NUMERO DE LINEA = ";LIN
2930 CLS
2940 FOR BU=1 TO W
2950   FOR X=1 TO 25
2960     PRINT
2970   NEXT
2980   LOCATE 1,1
2990   IF CANAL=8 THEN PRINT#8,PRINT#8,
3000   PRINT#CANAL,"VENTANA NUMERO: ";BU
3010   PRINT#CANAL,PRINT#CANAL,
3020   PRINT#CANAL,LIN;" LOCATE ";XPOR1(BU);";";YPOR1(BU);
3030   PRINT#CANAL,LIN+10;" PRINT CHR$(201);"
3040   PRINT#CANAL,LIN+20;" FOR X=";XPOR1(BU)+1;" TO ";XPOR2(BU)-1
3050   PRINT#CANAL,LIN+30;" PRINT CHR$(205);"
3060   PRINT#CANAL,LIN+40;" NEXT X"
3070   PRINT#CANAL,LIN+50;" PRINT CHR$(187);"
3080   PRINT#CANAL,LIN+60;" FOR Y=";YPOR1(BU)+1;" TO ";YPOR2(BU)-1
3090   PRINT#CANAL,LIN+70;" LOCATE ";XPOR1(BU);";";Y"
3100   PRINT#CANAL,LIN+80;" PRINT CHR$(186)"
3110   PRINT#CANAL,LIN+90;" LOCATE ";XPOR2(BU);";";Y"
3120   PRINT#CANAL,LIN+100;" PRINT CHR$(186)"
3130   PRINT#CANAL,LIN+110;" NEXT Y"
3140   PRINT#CANAL,LIN+120;" LOCATE ";XPOR1(BU);";";YPOR2(BU)
3150   PRINT#CANAL,LIN+130;" PRINT CHR$(200);"
3160   PRINT#CANAL,LIN+140;" FOR X=";XPOR1(BU)+1;" TO ";XPOR2(BU)-1
3170   PRINT#CANAL,LIN+150;" PRINT CHR$(205);"
3180   PRINT#CANAL,LIN+160;" NEXT X"
3190   PRINT#CANAL,LIN+170;" PRINT CHR$(188)"
3200   LET LIN=LIN+180
3210   LOCATE 25,32
3220   PRINT "(PULSA UNA TECLA)"
3230   LET A$=INKEY$
3240   IF A$="" THEN GOTO 3230
3250 NEXT BU
3260 CLS
3270 LET W=0
3280 RETURN
3290 REM
3300 REM *****
3310 REM ***** SUBROUTINA DE SELECCION DE OPCIONES *****
3320 REM *****
3330 REM
3340 LET OP=1
3350 LOCATE Y,X
3360 PRINT TS(N)
3370 LET A$=INKEY$
3380 IF A$="" THEN GOTO 3370
3390 IF A$=CHR$(18) THEN 3420
3400 IF A$="" THEN LET OP=-OP;LET N=N+1;IF N=3 THEN LET N=1
3410 GOTO 3350
3420 IF N=1 THEN GOSUB 2600
3430 IF N=2 THEN GOSUB 2700
3440 RETURN

```

■ Central de alarmas



ADA vez es más frecuente la necesidad de protección de los edificios para prevenir su deterioro por causas fortuitas o intencionadas. Los medios de que se dispone permiten instalar soluciones para la detección de

cualquier anomalía a los pocos instantes de producirse, con el consiguiente ahorro de las posibles pérdidas materiales y humanas.

En general, un sistema eficiente de detección de anomalías debe permitir, de una manera flexible, la captación de las variables ambientales y mediante su análisis determinar si el estado del entorno es el habitual o se han producido cambios. El interés por disponer de un sistema de captación de alarmas es múltiple: en primer lugar, la seguridad física de las personas y los bienes que se encuentran en el edificio; en segundo lugar, la prevención de robos u otro tipo de atentados, y en tercer lugar, puede servir para el control de la energía consumida. Los equipos necesarios para la medida y control son muy similares en las tres aplicaciones mencionadas y resultan fácilmente integrables en un mismo sistema mediante la utilización de un ordenador. Para los requerimientos de cálculo que estas aplicaciones necesitan, cualquier humilde ordenador personal es más que suficiente.

Vamos a describir los diferentes tipos de captadores necesarios para un sistema general de alarmas y mostraremos la forma más apropiada para su conexión a un ordenador personal cualquiera a través de las interfaces ya descritas del tipo de la tarjeta de ampliación de puertos de entrada/salida. Al utilizar un ordenador personal podremos diseñar pro-

gramas adaptados a cada necesidad y presentar el estado del sistema de la manera más adecuada a nuestras posibilidades. En la segunda parte veremos un ejemplo de programa que permita la utilización de los sensores descritos.

■ Sensores

Los fenómenos que interpretamos como alarmas pueden ser de muy diferentes magnitudes. Generalmente la producción de la alarma estará relacionada con el cambio de estado de una determinada magnitud, que será convertida en eléctrica a través de un sensor. Otras veces la producción de la alarma será consecuencia del paso de un valor de una magnitud física o química a un valor por encima o por debajo de un límite de normalidad. Cada magnitud será convertida en señal eléctrica a través de un sensor adecuado y normalizado su valor para poder ser comunicado al ordenador o equipo central de control. Algunos sistemas de protección podrían diseñarse de forma simplificada sin necesidad de ordenador, pero un sistema integral deberá contar con algún elemento lógico que le permita la toma de decisiones, al menos mientras se consigue la intervención de la persona o grupo interesados.

Según el tipo de magnitud captada por el sensor, distinguiremos las diferentes familias de sensores orientados a sistemas de seguridad:

- Electromecánicos: convierten una magnitud mecánica en eléctrica.

- Electromagnéticos: transforman radiaciones electromagnéticas en señales eléctricas.

EL TALLER DEL HARDWARE

— Electroquímicos: proporcionan señales eléctricas proporcionales a magnitudes químicas.

Pasemos a ver uno por uno los diferentes tipos de sensores y su conexión a un ordenador personal para crear un sistema de alarmas.

■ Sensores de apertura o cierre

Son los más sencillos de conectar, pues consisten simplemente en un circuito que se encuentra cerrado o abierto en posición normal y que generan alarma al pasar al estado contrario. El circuito eléctrico es el indicado en la figura.

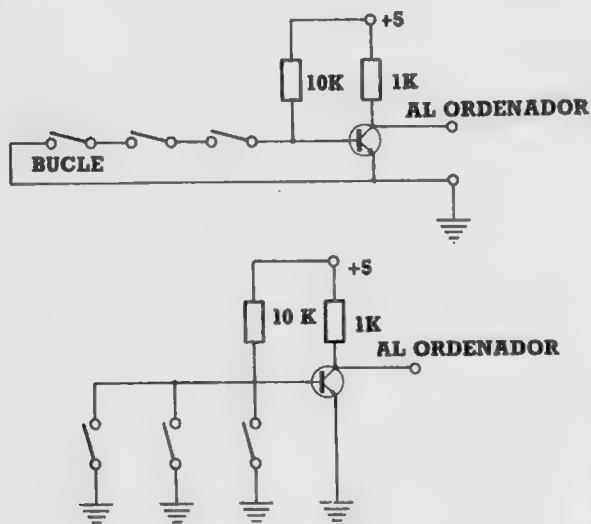


Fig. 1. Circuitos para sensor de apertura o cierre.

La salida se activa siempre que se produzca el cierre de uno de los contactos normalmente abiertos o la apertura de uno de los contactos normalmente cerrados. Es interesante hacer notar que los sensores de apertura se conectan en serie, mientras que los sensores de cierre se conectan en paralelo.

La forma de utilización es muy variada, existiendo contactos adaptados a infinidad de situaciones.

Para detección de apertura de puertas o ventanas se instala un contacto empotrado en la zona interior del marco y en la parte móvil otro dispositivo que cierre el contacto en posición de reposo. El contacto puede ser producido por un pulsador mecánico o, mejor todavía, por un imán sobre un relé de láminas.

Para la protección de ventanas acrista-

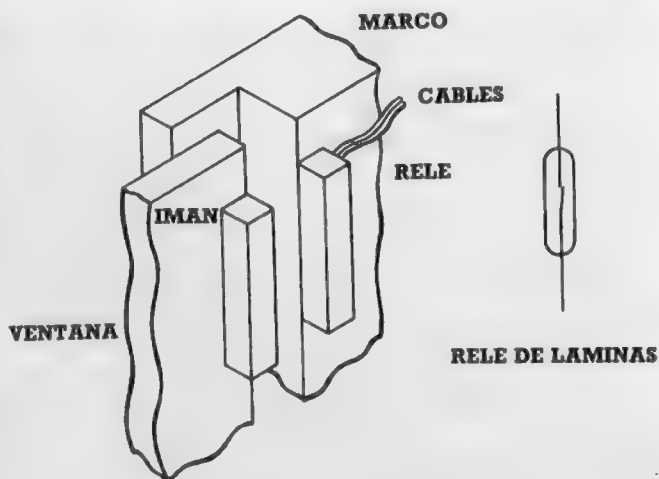


Fig. 2. Montaje de sensores de apertura o cierre.

ladas puede instalarse por todo el perímetro una tira adhesiva de metal que se rompería en caso de rotura del cristal, produciendo la apertura del circuito. Este tipo de sensores, por la naturaleza del elemento al que van asociados, son de "un solo uso", frente a los otros que puede probarse su funcionamiento sin que se deterioren.

Para poder determinar el lugar donde se produjo la alarma es necesario disponer de tantas entradas como áreas sensibles existan. Es conveniente no establecer un bucle de sensores muy largo, pues podrían sensibilizarse con señales que no fueran en realidad roturas, sino ruidos producidos por otros circuitos próximos.

Existen en el comercio diferentes tipos de malla autoadhesiva que se puede fijar a cualquier superficie susceptible de rotura y establecer un circuito con todos los puntos que interesa proteger.

Otra forma diferente de presentación es en forma de almohadilla, que al ser presionada cierra un contacto. Puede utilizarse para detectar el paso de una persona, situándola bajo una alfombra o moqueta de forma muy difícil de detectar.

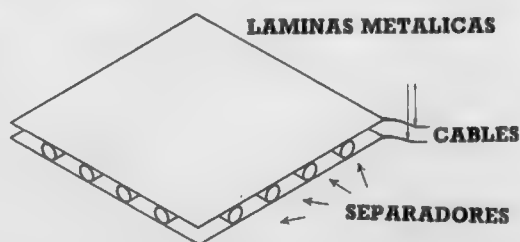


Fig. 3. Lámina de detección de paso.

Para la protección de un objeto valioso puede utilizarse una variante miniatura de la anterior, que, colocándose debajo, mantenga un contacto permanente en situación de reposo. Es más práctica la utilización de sensores con circuito abierto en reposo, pues no consumen nada.

■ Sensores de movimiento

Para detectar el ataque a un determinado objeto puede utilizarse un sensor que capte vibraciones o desplazamientos pequeños. El principio utilizado es el mismo que el de los sensores sísmicos: una masa suspendida de una lámina se encuentra entre varios contactos, alguno de los cuales se cerrará si el conjunto es sometido a movimientos.



Fig. 4. Sensor de movimiento con contactos.

Hay otras variantes de este mismo principio que realizan los contactos mediante una bola que se apoya en tres contactos. Al estar la bola en equilibrio poco estable, el más mínimo movimiento causa que alguno de los circuitos se rompa y, por tanto, se produzca la alarma. Estos tipos de sensores están sujetos a dar alarmas falsas si el lugar de instalación se encuentra cerca del paso normal de personas o vehículos que produzcan vibraciones similares a las que se intentan detectar.

Se pueden captar variaciones más pequeñas utilizando un método algo más complicado: la masa es un imán y se mueve en el interior de una bobina. Al desplazarse el imán en el interior de la bobina, en sus bornas se genera una tensión proporcional al desplazamiento y de frecuencia igual a la del movimiento. Con un amplificador puede conseguirse dar a la señal un valor directamente utilizable por el ordenador, pero resulta algo más

complejo que el anterior. A este tipo de sensor pertenecen los acelerómetros utilizados en ensayos sísmicos.

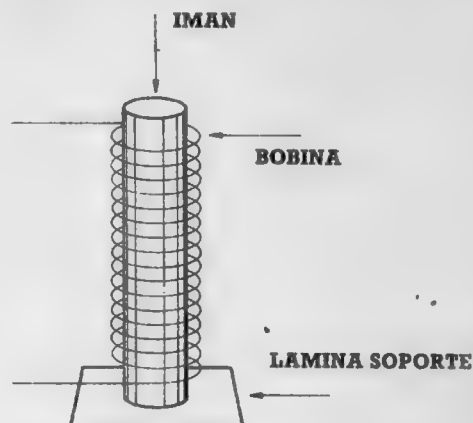


Fig. 5. Sensor electromagnético de movimiento.

■ Sensores térmicos

Una de las causas que se pretenden combatir es el fuego y cuanto antes mejor, por lo que es importante detectar cualquier elevación anómala de la temperatura en el ambiente. Las causas pueden ser naturales o de origen humano, por lo que deberán situarse los sensores próximos al punto donde se espera que pueda producirse la anomalía.

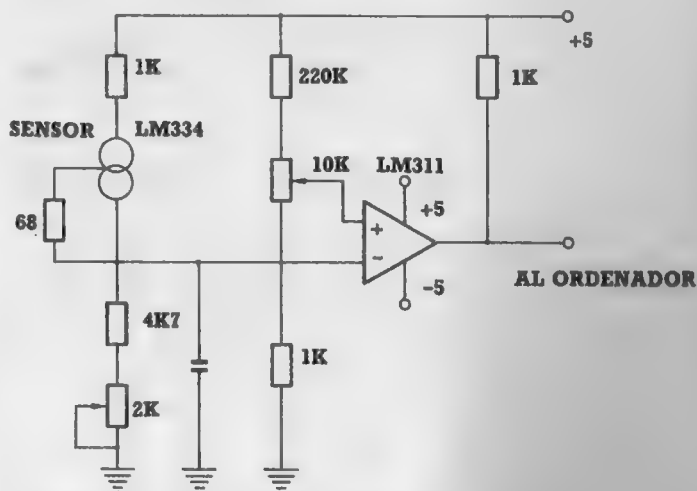


Fig. 6. Sensor proporcional de temperatura.

Los sensores de temperatura serán en general de tipo proporcional. Para conectarlos al sistema de alarmas podremos convertir la magnitud en una señal eléctrica proporcional, como ya se ha indicado en los ejemplos

EL TALLER DEL HARDWARE

de captación de señales analógicas. En los casos más frecuentes será suficiente proporcionar una señal binaria que indique si una determinada temperatura ha sido sobrepasada, disparándose entonces el actuador que contrarreste el efecto.



Fig. 7. Sensor de temperatura con lámina bimetálica.

En muchos sistemas de alarma y control de temperatura el sensor es simplemente una lámina de bimetálica que al sobrepasar una temperatura abre o cierra un circuito. El contacto puede usarse directamente como la señal de indicación de la alarma.

■ Barrera infrarroja

Es un tipo de barrera luminosa en el cual la luz utilizada no es visible, por lo que el intruso no puede ser consciente de que la rompe. Se suele montar con el emisor a un lado del paso que se desea proteger y el receptor en el contrario, si bien puede utilizarse la reflexión en un espejo para situar todos los componentes electrónicos en un mismo lado.

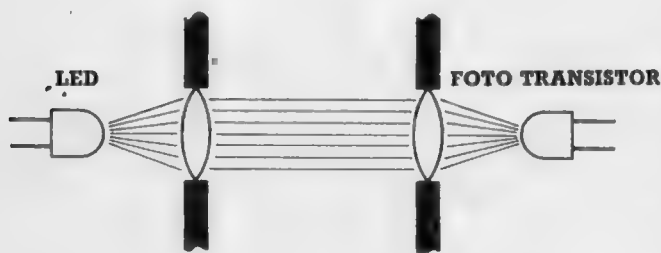


Fig. 8. Barrera de infrarrojos.

Para que el sistema resulte más robusto se suele emplear luz modulada a baja frecuencia, detectando la señal reflejada de forma síncrona, con lo que se consigue un mayor rechazo a las señales externas y a los posibles ataques con una linterna, por ejemplo.

La barrera puede realmente cubrir una gran superficie, mediante múltiples reflexio-

nes en espejos situados en el marco de la zona. El problema es que también resultan más sensibles a los fenómenos atmosféricos y suciedad en las superficies reflectantes.

El circuito electrónico necesario para la captación de la alteración de la barrera es muy simple:

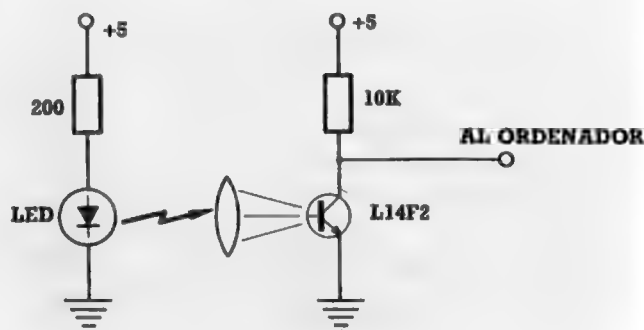


Fig. 9. Circuito de la barrera de infrarrojos.

La señal suministrada es similar a la descrita en la utilización de sensores ópticos, siendo necesaria tanto mayor amplificación cuanto mayor sea la distancia a cubrir con la protección.

■ Detector de infrarrojos

Una manera muy interesante de detectar la presencia de personas en un recinto es mediante la captación de su radiación infrarroja global. La radiación es realmente muy baja, pero puede amplificarse suficientemente para poder ser detectada con los medios habituales. Lo que normalmente interesará es detectar las variaciones producidas de forma brusca en un determinado espacio y que será debida a la presencia de personas. Puesto que la radiación media será proporcional a la temperatura de las paredes y ésta variará con la hora del día será necesario dotar al sensor de una discriminación para que no resulte sensible a las variaciones muy lentas. Un circuito pasa-altos de frecuencia de corte bastante baja resolverá el problema.

El sensor deberá ser colocado en un lugar que permita cubrir la zona de protección con el ángulo de visión propio del sensor. Si se desea aumentar la sensibilidad al movimiento de personas, puede cubrirse el sensor con una rejilla, que establezca un diagrama de

sensibilidad en forma de sierra, por lo que una persona en movimiento producirá una gran se-

ñal, aunque la cantidad global emitida sea la misma en todas sus posiciones.

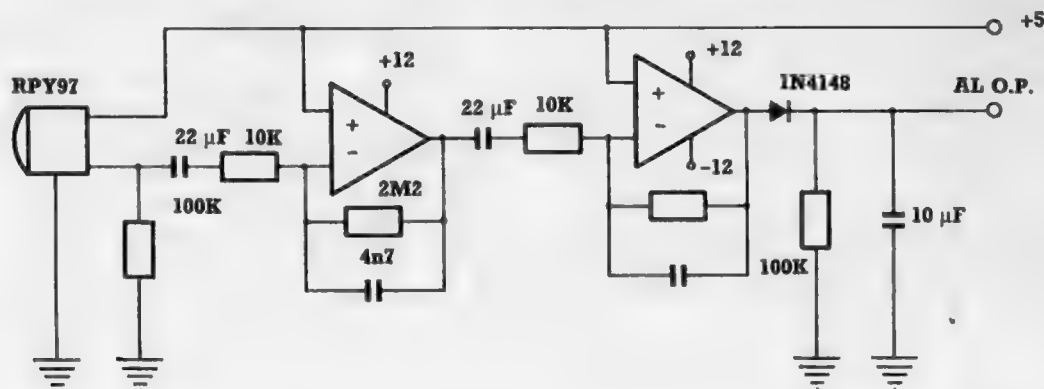


Fig. 10. Detector de infrarrojos.

El circuito es básicamente un amplificador de alterna conectado a un fototransistor sensible en la banda de infrarrojos. La señal resultante es amplificada en alterna y rectificada, para dar una resultante en nivel lógico indicativa de que hay intruso o no.

En este tipo de sensores es todo el conjunto de cosas que se encuentran delante las que pueden causar la alarma, por lo que a veces se les denomina volumétricos. El efecto se acentúa más si en las paredes opuestas al sensor se colocan superficies reflectantes a la radiación empleada, de forma que se produzcan efectos de modulación al paso del posible intruso, con el consiguiente aumento de sensibilidad.

Los circuitos comerciales se basan en equipos que utilizan diodos de efecto Gunn para producir la oscilación de microondas, generar una señal de algunos milivatios y captar la señal refleja. El resultado a la salida del oscilador-sensor es una señal de baja frecuencia proporcional a la velocidad del movimiento del objeto detectado. Son extremadamente sensibles y es conveniente utilizarlo con el equipo que acompaña al oscilador. No son de fácil adquisición en nuestro mercado como componente independiente, por lo que lo citamos por existir como conjunto en muy diversas aplicaciones. Una de las más frecuentes es la medida de la velocidad de vehículos en carretera.

■ Detectores de microondas

Siguiendo con el mismo procedimiento que el empleado en los sensores de infrarrojos pero utilizando una frecuencia en la banda X, es decir, del orden de 10 GHz., utilizan el efecto Doppler. Este conocido efecto consiste en que la señal reflejada por un objeto en movimiento es de frecuencia diferente a la de la señal emitida, siendo mayor si el objeto se acerca y menor si se aleja. En los sistemas de seguridad permite conocer el sentido del movimiento de la persona que se detecta.

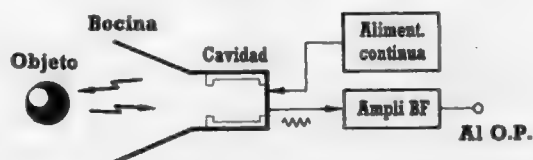


Fig. 11. Detector de microondas. Esquema de bloques.

■ Detector de ultrasonidos

Con un principio similar al anterior, pero utilizando ultrasonidos, se pueden emplear para detectar movimientos de objetos o personas. La frecuencia a la que se trabaja es mucho menor, de 20 a 40 KHz. Los transductores utilizados como emisor y receptor se emplean como altavoces y micrófonos, pero con las dimensiones adaptadas a las frecuencias de trabajo.

Las señales producidas en el receptor están muy expuestas a ruidos del ambiente, por lo que es necesario realizar un filtrado y tratamiento de la señal, para poder utilizarlos en la práctica. Los que se utilizan actualmente son tan fiables como los de tipo microondas, con los que se utiliza a veces en sensores combinados.

■ Detector de humos

En muchas ocasiones los elementos que entran en combustión no originan llamas, como, por ejemplo, algunos cables y determinados tipos de plásticos. Sin embargo, producen cantidad de gases que presentan una opacidad detectable físicamente o reaccionan con sensores de forma química. En la figura se muestra un circuito detector de humos. El circuito está diseñado para producir una corriente suficiente para activar un relé si se detecta la presencia de humo.

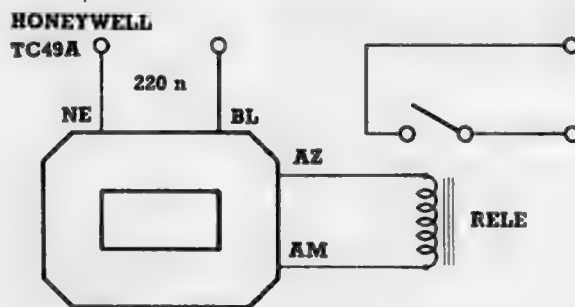


Fig. 12. Detector de humos.

■ Detector de humedad

En nuestras casas actuales, cada vez más automatizadas, es frecuente que alguno de los electrodomésticos deje escapar el agua, ocasionando molestias y deterioros en la propia

casa y en las inferiores. Por esta y otras razones es conveniente detectar la presencia de humedades excesivas en diferentes lugares. El sensor consiste en un circuito que al resultar humedecido varía su conductividad, pudiendo amplificarse la variación hasta hacerla compatible con el ordenador.

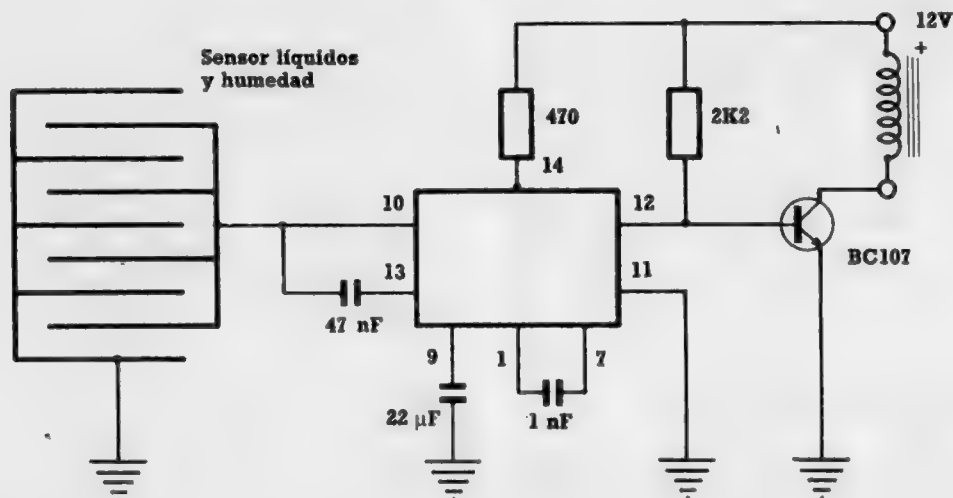


Fig. 13. Detector de humedad.

■ Detector acústico

Es una entrada más a tener en cuenta a la hora de determinar la presencia de intrusos en un lugar. Cuando es necesario detectar un ruido sospechoso y distinguirlo del resto de señales que llegan al micrófono es necesario tratar la señal de forma conveniente para separar las diferentes causas. En primer lugar, sería conveniente que el micrófono fuera de la direccionalidad adecuada al local, para

poder diferenciar el ruido en condiciones de reposo del que se presenta en condiciones de intrusión.

La curva de respuesta deberá ser muy específica para adaptarse a las posibles causas. Es necesario, además, tener en cuenta la amplitud y duración de las señales a detectar. La señal debe contemplarse de forma adaptativa, pues las condiciones del entorno pueden variar de un momento a otro, sin que deba interpretarse siempre como causa de alarma.

El circuito de amplificación y detección de nivel será igual que los empleados en otros tipos de sensores.

■ **Detector sísmico**

Su empleo principal es para prevenir desastres naturales, pero es de aplicación también en seguridad de recintos, para detectar la más mínima vibración producida por elementos naturales o por intervención humana.

El campo donde más se ha empleado es en prospección geológica, determinando el efecto de un estímulo de choque en numerosas sondas colocadas sobre el terreno, en for-

ma de rejilla. El tratamiento por ordenador de las señales recibidas permite determinar con exactitud en qué puntos se producen reflexiones bajo la corteza terrestre y, por tanto, dónde puede haber un cambio de las características del terreno.

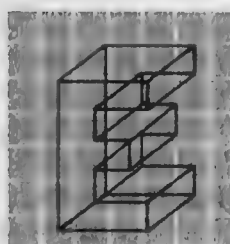
■ **Conclusión**

Los sensores reseñados ofrecen una salida binaria indicativa de su estado, bien directamente o después de ser procesados por un circuito electrónico simple. Para su integración en un sistema basado en ordenadores personales veremos la forma de conectarlos y utilizarlos por programa.

APRENDER CON EL ORDENADOR

NATURALEZA Y TECNOLOGIA

■ Manejo y equivalencias de las distintas unidades de fuerza, trabajo y potencia



Numerosos problemas de física nos encontramos con que los datos que se utilizan vienen en unidades distintas y muchas veces resulta complicado conseguir ponerlos todos en las mismas unidades. El programa que

se ofrece a continuación nos proporciona las principales unidades de fuerza, trabajo y potencia, sus equivalencias y nos pasa de unas unidades a otras de una manera muy sencilla.

A continuación se ofrece un cuadro con las unidades que maneja el programa y sus equivalencias:

Magnitud	Sistema terrestre	Sistema internacional	Sistema CGS	Otras unidades	Equivalencias
FUERZA	Kilopondio (Kp)	Newton (Nw)	Dina	—	1 Kp = 9,8 Nw 1 Nw = 10 ⁵ dinas
TRABAJO	Kilopondímetro (Kpm)	Julio (J)	Ergio	Kilowatio x hora (Kw x h)	1 Kpm = 9,8 julios 1 julio = 10 ⁷ ergios 1 Kw x h = 3,6 · 10 ⁶ J.
POTENCIA	Kilopondímetro/sg	Watio (W)	Ergio/sg	Caballo de vapor (CV)	1 CV = 75 Kpm/sg. 1 CV = 735 W 1 Kp/sg = 9,8 W

Fig. 1.

```

10 REM *****
20 REM **
30 REM **      PROGRAMA PARA EL MANEJO Y TRANSFORMACION DE UNIDADES
40 REM **      DE FUERZA, TRABAJO Y POTENCIA
50 REM **
60 REM *****
70 REM *****
80 REM INTRODUCCION DE DATOS
90 REM *****
100 DIM F$(4):DIM T$(4):DIM P$(4):DIM N$(4)
110 DIM C1(4):DIM C2(4):DIM C3(4):DIM C(4)
120 DATA "KILOPONDIO",9.8,"NEWTON",100000,"DINA",1,"DINA",1.020408E-6
130 DATA "KILOPONDIMETRO",9.8,"JULIO",10000000,"ERGIO",2.777777E-14
140 DATA "KILOWATIO x HORA",367347

```

```

150 DATA "KILOPONDIMETRO/SG",9.8,"WATIO",10000000,"ERGIO/SG",1.358696E-10
160 DATA "CABALLO DE VAPOR",75
170 FOR I=1 TO 4 :READ F$(I):READ C1(I):NEXT I
180 FOR I=1 TO 4 :READ T$(I):READ C2(I):NEXT I
190 FOR I=1 TO 4 :READ P$(I):READ C3(I):NEXT I
200 CLS
210 REM *****
220 REM MENU PRINCIPAL
230 REM *****
240 PRINT :PRINT "MAGNITUDES:"
250 PRINT :PRINT "FUERZA      = Masa x aceleracion  ---X1"
260 PRINT :PRINT "TRABAJO      = Fuerza x longitud    ---X2"
270 PRINT :PRINT "POTENCIA     = Trabajo / tiempo      ---X3"
275 PRINT :PRINT "TERMINAR
280 PRINT :INPUT "ELIGE OPCION : ",OP
290 CLS
300 REM *****
310 REM REALIZACION DE LAS OPCIONES DEL MENU PRINCIPAL
320 REM *****
330 ON OP GOTO 350,550,750,950
350 PRINT :PRINT "UNIDADES DE FUERZA : "
360 FOR I=1 TO 4 :LET N$(I)=F$(I):LET C(I)=C1(I):NEXT I
370 GOSUB 1000
380 PRINT " 1 Kilopondio = 9.8 Newton "
390 PRINT " 1 Newton    = 100000 Dinas"
400 PRINT :PRINT "FORMULAS:"
410 PRINT " 1 Kilopondio= 9.8 Kg(masa) x 1 m/sg2 "
420 PRINT " 1 Newton    = 1 Kg (masa) x 1 m/sg2"
430 PRINT " 1 Dina      = 1 gr  x 1cm/sg2"
440 PRINT
450 PRINT "Nota: El Kilopondio tambien recibe el nombre de kilogramo-peso"
460 GOSUB 2000
470 GOTO 200
550 PRINT :PRINT "UNIDADES DE TRABAJO : "
560 FOR I=1 TO 4:LET N$(I)=T$(I):LET C(I)=C2(I):NEXT I
570 GOSUB 1000
580 PRINT " 1 Kilopondimetro = 9.8 Julios "
590 PRINT " 1 Julio          = 10000000 Ergios "
600 PRINT " 1 kilowatio x hora = 3600000 Julios"
610 PRINT :PRINT "FORMULAS:"
620 PRINT " 1 Kilopondimetro = 1 Kilopondio x 1 metro"
630 PRINT " 1 Julio          = 1 Newton    x 1 metro"
640 PRINT " 1 Ergio          = 1 Dina      x 1 cm  "
650 PRINT
660 GOSUB 2000
670 GOTO 200
750 PRINT :PRINT "UNIDADES DE POTENCIA : "
760 FOR I=1 TO 4:LET N$(I)=P$(I):LET C(I)=C3(I):NEXT I
770 GOSUB 1000
780 PRINT " 1 Kilopondimetro/sg = 9.8 Watios"
790 PRINT " 1 Watio             = 10000000 Ergios/sg"
800 PRINT " 1 Caballo de vapor = 75 Kilopondimetro/sg"
810 PRINT " 1 Caballo de vapor = 736 Watios"
820 PRINT :PRINT :PRINT "FORMULAS : "
830 PRINT " 1 Watio = 1 Julio /sg"
840 PRINT
850 GOSUB 2000
860 GOTO 200
950 END
960 REM *****
970 REM SUBROUTINA PARA ESCRIBIR LAS UNIDADES EN LOS DISTINTOS SISTEMAS
980 REM *****
1000 PRINT "SISTEMA TERRESTRE      : ";N$(1)
1010 PRINT "SISTEMA INTERNACIONAL   : ";N$(2)
1020 PRINT "SISTEMA CGS              : ";N$(3)
1030 IF OP <> 1 THEN PRINT "OTRA MEDIDA          : ";N$(4)
1040 PRINT "EQUIVALENCIAS:"
1050 RETURN
1060 REM *****
1070 REM SUBROUTINA PARA TRANSFORMAR UNAS UNIDADES EN OTRAS
1080 REM *****
2000 PRINT :INPUT "¿QUIERES REALIZAR ALGUNA TRANSFORMACION(S/N)";R$
2010 IF R$="N" THEN RETURN
2020 CLS
2030 LET LIM=4
2040 IF OP=1 THEN LET LIM=3
2050 FOR I = 1 TO LIM
2060 PRINT N$(I);TAB(25);"  --->";I
2070 NEXT I
2080 INPUT "INTRODUCE EL NUMERO DE LA UNIDAD QUE QUIERES TRANSFORMAR ";UNI
2090 PRINT "¿CUANTOS ";N$(UNI);" QUIERES TRANSFORMAR?";INPUT CAN
2100 PRINT :PRINT :LET CAN=CAN:LET ACT=UNI
2110 FOR I=1 TO 3

```

```

2120 LET CANT=CANT*(C(ACT))
2130 LET SIG=(ACT MOD 4)+1
2140 IF OP=1 AND ACT=3 THEN GOTO 2160
2150 PRINT CANIN*(UN1); " SON "; CANTIN*(SIG)
2160 LET ACT=SIG
2170 NEXT I
2180 PRINT "PULSA UNA TECLA PARA VOLVER AL MENU"
2190 LET A$=INKEY$
2200 IF A$="" THEN GOTO 2190
2210 RETURN
3000 REM *****
3010 REM ** MODIFICACIONES PARA SPECTRUM **
3020 REM ** 100 DIM F$(4,10):DIM T$(4,16):DIM P$(4,17):DIM N$(4,17) **
3030 REM ** 330 GOTO 150+(200*OP) **
3040 REM ** 950 GOTO 9999 **
3050 REM ** MODIFICACIONES PARA COMMODORE **
3060 REM ** Cambiar CLS por PRINT CHR$(147) **
3070 REM ** 2190 GET A$ **
3080 REM *****

```

Comentario del programa

Cuando damos al programa la orden de ejecución, en primer lugar nos saca por pantalla un menú principal en el que se ofrecen las tres magnitudes (fuerza, trabajo y potencia) y se indica que se seleccione una de ellas. Cuando se selecciona una de ellas, aparecen en pantalla las unidades de ésta en los distintos sistemas y sus equivalencias y a continuación se pregunta si se quiere realizar alguna transformación. En caso afirmativo se pregunta qué unidad se quiere transformar y qué cantidad de ésta, apareciendo a continuación las equivalencias al dato introducido en el resto de las unidades.

En las siguientes figuras se ofrece cómo queda la pantalla cuando aparece el menú principal, las unidades y equivalencias de una magnitud y cuando se realiza un cambio de una unidad a otras.

```

MAGNITUDES:

FUERZA      = Masa x aceleracion  ---->1
TRABAJO      = Fuerza x longitud   ---->2
POTENCIA     = Trabajo / tiempo    ---->3
TERMINAR     .                    ---->4

ELIGE OPCION :

```

Fig. 2.

El programa tiene cinco partes diferenciadas. En la primera se introducen los datos. El programa va a utilizar tres listas (DIM) alfa-

```

UNIDADES DE TRABAJO :
SISTEMA TERRESTRE   : KILOPONDIMETRO
SISTEMA INTERNACIONAL : JULIO
SISTEMA CGS          : ERGIO
OTRA MEDIDA         : KILOMATIO x HORA
EQUIVALENCIAS:
1 Kilopondimetro    = 9.8 Julios
1 Julio             = 1000000 Ergios
1 Kilomatio x hora  = 360000 Julios

FORMULAS:
1 Kilopondimetro    = 1 Kilopondio x 1 metro
1 Julio             = 1 Newton x 1 metro
1 Ergio             = 1 Dina x 1 cm

¿QUIERES REALIZAR ALGUNA TRANSFORMACION(S/N)?

```

Fig. 3.

```

KILOPONDIMETRO      ----> 1
JULIO                ----> 2
ERGIO                ----> 3
KILOMATIO x HORA     ----> 4
INTRODUCE EL NUMERO DE LA UNIDAD QUE QUIERES TRANSFORMAR 2
¿CUANTOS JULIO QUIERES TRANSFORMAR? 50

50 JULIO SON 5E+08 ERGIO
50 JULIO SON 1.388889E-05 KILOMATIO x HORA
50 JULIO SON 5.10204E-01 KILOPONDIMETRO
PULSA UNA TECLA PARA VOLVER AL MENU

```

Fig. 4.

numéricas (F\$, T\$ y P\$) para guardar los nombres de las unidades de fuerza, trabajo y potencia, respectivamente. También usa tres listas numéricas (C1, C2 y C3) para guardar los coeficientes por los que hay que multiplicar una unidad para pasar a la siguiente. Así, en C1(1) estará el coeficiente por el que hay que

multiplicar la unidad de fuerza guardada en F\$(1) (kilopondio) para pasarla a Newton (unidad guardada en F\$(2)). En C1(2) se encontrará el coeficiente por el que hay que multiplicar los Newton para pasarlos a Dinás (unidad guardada en F\$(3)). En C1(4) se encontrará el coeficiente por el que hay que multiplicar la última unidad para pasarla a la primera. De la misma manera en C2 están los coeficientes de transformación de las unidades de trabajo y en C3 los de las unidades de potencia. Notar que como únicamente hay tres unidades de fuerza y para guardar la uniformidad de cuatro unidades por magnitud para que luego resulte más sencillo en operaciones posteriores, en F\$(3) y en F\$(4) se ha guardado la misma unidad (dina) habiéndose puesto lógicamente como coeficiente de transformación el 1. Todos estos datos se introducen en el programa mediante instrucciones DATA y a continuación se leen y se guardan en las listas correspondientes.

La segunda parte del programa es la encargada de visualizar el menú principal y solicitar la opción.

A continuación, dependiendo de la opción elegida, la instrucción 330 selecciona la línea a la que se debe saltar y a partir de cada una de éstas se realizan las operaciones indicadas anteriormente. Para realizar estas operaciones se utilizan dos subrutinas. La subrutina 1000 escribe las unidades de los distintos sistemas correspondientes a la magnitud seleccionada. Para ello antes de llamar a las subrutinas se guardan en las listas N\$ y C los nombres y los coeficientes, respectivamente, correspondientes a la magnitud seleccionada, ya que las subrutinas utilizan estas listas para dichos datos. La subrutina 2000 es la encargada de transformar la unidad introducida en el resto de las unidades de la misma magnitud. Después de pedir los datos, el FOR de las instrucciones 2110-2170 es el que realiza la transformación. El significado de las variables utilizadas en este FOR es el siguiente:

UNI → Número de unidad que se quiere transformar según el orden establecido al principio.

CAN → Cantidad de esta unidad.

CANT → Cantidad equivalente de las distintas unidades que el programa va a ir transformando en cada momento.

ACT → Número de la unidad que va a transformar el programa en un momento determinado.

SIG → Número de la siguiente unidad a la que está transformando el programa en un momento determinado.

Notar que en las dos subrutinas aparecen instrucciones IF en las que se pregunta si OP=1, ya que en este caso estaremos tratando las unidades de fuerza, de las que ya hemos dicho anteriormente que sólo hay tres, por lo que no se deberá escribir la unidad número 4.

MATEMATICAS

■ Cambios de base

Una aplicación matemática que suele aparecer en muchas ocasiones y sobre todo en numerosos problemas relacionados con la informática es la del cambio de base. En esta sección se ofrece a continuación un sencillo programa que realiza la transformación desde base 10 a cualquier otra base inferior a 10, y viceversa.

Seguidamente vemos la fórmula general para pasar desde una base inferior a 10 a base 10:

$$d_1 d_2 d_3 \dots d_n \text{ dígitos del n.º en base menor que 10.}$$

$$N = \sum_{i=1}^n d_i \cdot B^{n-1} \text{ (siendo B la base de numeración del n.º inicial).}$$

Fig. 5.

Esta fórmula se aplicaría a un ejemplo de la siguiente manera:

Sea 11001 en base 2:

$$\begin{array}{rcl}
 1 & 1 & 0 & 0 & 1 \\
 \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
 1 & \cdot 2^4 & = & 16 \\
 0 & \cdot 2^3 & = & 0 \\
 0 & \cdot 2^2 & = & 0 \\
 1 & \cdot 2^1 & = & 2 \\
 1 & \cdot 2^0 & = & 1 \\
 \hline
 & & & & 25
 \end{array}$$

Fig. 6.

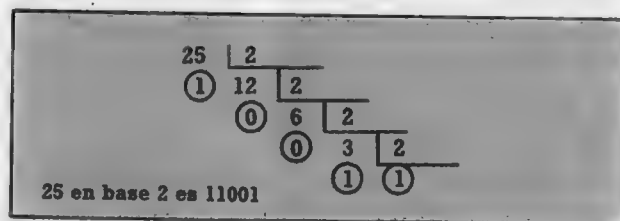
Para pasar desde base 10 a una base inferior a 10, lo que hay que hacer es dividir el número que queremos pasar entre la base de manera sucesiva hasta que el cociente que nos quede sea menor que la base. Entonces, el resultado será el número cuyo primer dígito es el último cociente obtenido y el resto de los dí-

APRENDER CON EL ORDENADOR

gitos son los restos de las sucesivas divisiones realizadas tomados en orden contrario a aquel en el que se han realizado las divisiones.

En el siguiente ejemplo podemos ver cómo se aplica este método:

Fig. 7.



```

10 REM *****
20 REM **
30 REM **          PROGRAMA PARA REALIZAR CAMBIOS DE BASE
40 REM **          DE BASE 10 A CUALQUIER OTRA Y VICEVERSA
50 REM **
60 REM *****
70 CSD
80 PRINT "MENU:"
90 PRINT "PASAR DE UNA BASE MENOR QUE 10 A BASE 10"
100 PRINT "PASAR DE BASE 10 A UNA BASE MENOR QUE 10"
110 PRINT "INTRODUCE OPCION ";OP
120 CLS
130 ON OP GOTO 140,260
140 INPUT "EN QUE BASE VAS A INTRODUCIR EL NUMERO ";N
150 IF N<2 OR N>9 THEN GOTO 140
160 PRINT "INTRODUCE EL NUMERO ";N$
170 R=0
180 FOR I=LEN(N$) TO 1 STEP -1
190 L=MID$(N$,I,1)
200 L=VAL(L)
210 R=R*(L*(N*(LEN(N$)-1)))
220 NEXT I
230 PRINT "PRINT";PRINT
240 PRINT N$," EN BASE 10 ES ";R
250 END
260 INPUT "INTRODUCE EL NUMERO QUE QUIERES TRANSFORMAR ";NUM
270 PRINT "A QUE BASE 10 QUIERES PASAR ";N
280 IF N<2 OR N>9 THEN GOTO 270
290 LET CIF=1:LET X=NUM
300 IF X<N THEN GOTO 340
310 LET X=X/N
320 LET CIF=CIF+1
330 GOTO 300
340 DIM B(CIF):LET X=NUM
350 FOR I=CIF TO 2 STEP -1
360 LET B(I)=X MOD N
370 LET X=X/N
380 NEXT I
390 LET B(1)=X
400 PRINT "PRINT";PRINT
410 PRINT NUM," EN BASE ";N," ES"
420 FOR I=1 TO CIF
430 PRINT B(I);
440 NEXT I
450 REM *****
460 REM ** MODIFICACIONES PARA EL COMMODORE
470 REM ** Cambiar CLS por PRINT CHR$(147) en 20 y 120
480 REM **
490 REM ** MODIFICACIONES PARA SPECTRUM
500 REM ** Para el SPECTRUM solo es valida la opcion de pasar un nu
510 REM ** de base 10 a otra base, por lo que unicamente se deben po
520 REM ** ner las instrucciones 180 y de 120 hasta el final
520 REM *****

```

Comentario del programa

El programa comienza exponiendo un menú en el que se dan las dos posibles opciones y se solicita que se seleccione una de

ellas. La instrucción 130 produce un salto a la línea 140 ó 260, según se haya dado una opción u otra.

Entre las instrucciones 140 y 250 se realizan las operaciones de paso de una base me-

nor que 10 a base 10. El número que se quiere pasar se guarda en la variable N\$ y el resultado se guardará en R. El FOR comprendido entre 180 y 220 es el encargado de realizar las operaciones correspondientes a la fórmula expresada anteriormente: obtiene cada dígito mediante la función MID\$, pasa este dígito a una variable numérica y aplica la fórmula correspondiente al dígito sumando el resultado de esta operación a la variable R, que es en la que vamos acumulando el resultado.

Las instrucciones comprendidas entre 260 y 440 son las que realizan la operación contraria. Entre la 290 y la 330 se calcula el número de cifras que va a tener el número resultante. Para ello, se van realizando divisiones enteras consecutivas hasta que el cociente (guardado en X) sea menor que la base (guardada en N). Cada vez que se realiza una de estas divisiones se suma 1 a la variable CIF, que es aquella en la que vamos a guardar el número de cifras del número que queremos hallar. Esta variable se inicializa a 1, ya que el número total de cifras será igual al número de divisiones realizadas más una (la correspondiente al último cociente).

A continuación se declara en 340 la lista (DIM) B, en cuyos elementos se guardarán cada una de las cifras del número resultante. De ahí que antes de declarar esta lista necesitemos conocer el número de cifras de que va a constar dicho número.

El FOR de las instrucciones 350-380 es el encargado de hallar los dígitos del número buscado. Para ello, va realizando las divisiones consecutivas y guardando en cada lugar correspondiente el resto de estas divisiones. La variable del FOR comienza en la última cifra y termina en el 2, porque el resto de la primera división realizada será el último dígito, y así sucesivamente hasta el resto de la última división, que será el segundo dígito.

La instrucción 390 guarda el último cociente como primer dígito y en 420-440 se escribe el número resultante.

NOTA: Como ya se ha indicado en el listado, la opción de pasar de una base menor que 10 a base 10 no es válida para el SPECTRUM, ya que en este ordenador no existe la función MID\$. Se deja como ejercicio a algún avisado usuario de este equipo que intente realizar esta opción con otras estructuras más complicadas, como pueden ser los DIM.

SOCIEDAD

■ Test de inglés

A continuación se ofrece un programa con el que vas a tener la oportunidad de aprender distintas palabras y estructuras del inglés a través de un entretenido juego.

El programa, después de darle la orden de ejecución, escribirá en pantalla una frase en inglés en la que faltará una palabra o un grupo de palabras consecutivas. En el lugar donde falten éstas se escribirán puntos suspensivos. A continuación, escribirá también la primera y la última letra de lo que falta en la frase, así como tantos espacios entre éstas como letras falten. Cuando falte más de una palabra, también colocará un espacio por cada espacio de separación entre las palabras. Seguidamente, tú deberás ir tecleando las letras que creas que faltan para completar la palabra o palabras omitidas en la frase. Cuando te clees una letra correcta, el programa te la colocará en su lugar y así deberás continuar hasta que consigas completar todas las letras que faltan. Los espacios que aparezcan y que correspondan a los espacios en blanco de separación entre las palabras no debes completarlos evidentemente. En la siguiente figura podemos observar cómo tendríamos la pantalla en un momento determinado de la ejecución mientras estamos intentando adivinar la palabra o palabras:



Fig. 8.

En esta otra figura vemos lo que nos aparecería cuando ya hemos acertado todas las letras que faltaban:

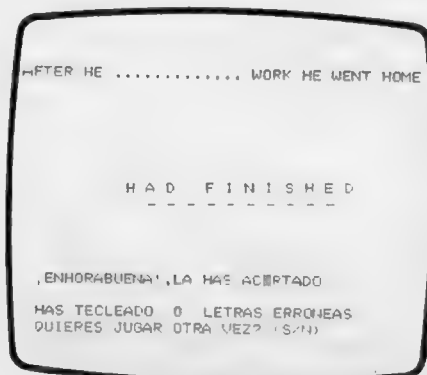


Fig. 9.

Comentario del programa

El esquema general del programa es el mismo que el de otros programas basados en el mismo juego publicados en números anteriores, variando en este caso los datos correspondientes a las frases y palabras que hay que adivinar.

```

10 REM *****
20 REM **
30 REM ** TEST DE INGLES **
40 REM **
50 REM *****
60 DIM P$(20,14):DIM L$(14)
70 REM LA INSTRUCCION 100 SOLO ES NECESARIO UTILIZARLA PARA MS-DOS (IBM O
   COMPATIBLES)
80 RANDOMIZE TIMER
90 DATA 84,79,79,75,42,79,70,70,0,76,79,79,75,42,65,84,0,65,77,42,76,79,79,75
100 DATA 78,78,71,42,70,79,82,0,76,79,79,75,42,65,70,84,69,82,0,74,85,83,84,42
110 DATA 65,83,0,72,65,68,42,83,69,84,0,72,65,68,42,70,73,78,73,83,72,69,68,0
120 DATA 72,65,86,69,42,84,73,77,69,0,79,78,42,67,89,67,76,73,78,71,0
130 DATA 65,42,76,79,84,0,65,84,42,84,72,69,0,84,79,42,66,69,0,87,65,83,42,71
140 DATA 78,86,69,78,0,83,72,65,76,76,0,65,82,82,73,86,69,83,0,87,72,69,78,0
150 DATA 83,72,79,85,76,68,0,84,79,42,77,65,75,69,0,68,82,69,87,42,66,65,67,75
155 DATA 0,77,73,78,68,0
160 PRINT "PRINT ",UN MOMENTO!,ESTOY CARGANDO LOS DATOS"
170 FOR I=1 TO 20
180 LET J=1
190 READ A
200 IF A=0 THEN FOR K=J TO 14 :LET P$(I,K)=" ":NEXT K :GOTO 230
210 LET P$(I,J)=CHR$(A)
220 LET J=J+1:GOTO 190
230 NEXT I
240 LET PAL=INT(RND(1)*20)+1
250 FOR J=1 TO 14 :LET L$(J)=" ":NEXT J
260 LET ETHO:LET CON=0
270 CLS
280 ON PAL GOTO 290,300,310,320,330,340,350,360,370,380,390,400,410,420,430,440,
   450,460,470,480
290 PRINT "IT WAS VERY HOT, SO I ..... MY COAT":GOTO 490
300 PRINT "COME AND ..... MY PHOTOGRAPH ALBUM":GOTO 490
310 PRINT "I ..... MY PEN. I LOST IT THIS MORNING":GOTO 490
320 PRINT "WILL YOU ..... THE CHILDREN FOR ME PLEASE?":GOTO 490
330 PRINT "SOMEONE KNOCKED AT THE DOOR ..... I WAS GETTING INTO THE BATH":
   GOTO 490
340 PRINT "AS SOON AS THE SUN ..... WE RETURNED TO OUR HOTEL":GOTO 490
350 PRINT "AFTER HE ..... WORK HE WENT HOME":GOTO 490
360 PRINT "IF I ..... I SHALL BE WRITING TO HIM TOMORROW":GOTO 490
370 PRINT "I AM VERY KEEN .....":GOTO 490
380 PRINT "THERE IS NOT ..... I CAN DO TO HELP HIM":GOTO 490
390 PRINT "A LARGE CROWD WAS WAITING ..... BUS-STOP":GOTO 490
400 PRINT "HE IS SAID ..... A GENIUS":GOTO 490
410 PRINT "A PEN ..... TO ME":GOTO 490
420 PRINT "IF IT RAINS TOMORROW WE ..... STAY AT HOME":GOTO 490
430 PRINT "THE MOMENT HE ..... I SHALL LET YOU KNOW":GOTO 490
440 PRINT "I SHALL GIVE HIM THE MESSAGE ..... HE RETURNS":GOTO 490
450 PRINT "SHE INSISTED THAT I ..... STAY TO LUNCH":GOTO 490
460 PRINT "SHE TOLD THEM NOT ..... SO MUCH NOISE":GOTO 490
470 PRINT "THE DOG ..... IN TERROR WHEN IT SAW THE SNAKE":GOTO 490
480 PRINT "I DO NOT ..... WAITING"
490 LET L$(1)=P$(PAL,1):LET ULT=1
500 FOR I=14 TO 2 STEP-1
510 IF P$(PAL,I)<>" " AND ULT=1 THEN LET L$(I)=P$(PAL,I):LET F=I:LET ULT=0
515 IF P$(PAL,I)="*" THEN LET L$(I)="*"
520 NEXT I
530 LOCATE 10,10
540 FOR I=1 TO F
545 IF L$(I)="" THEN PRINT " ":GOTO 560

```

```

550 PRINT L$(I);
560 NEXT I
570 PRINT:PRINT TAB(12);
580 FOR I=2 TO F-1
590 PRINT "- ";
600 NEXT I
610 REM PARA AMSTRAD :560 LOCATE 1,15
620 LOCATE 15,1
630 IF FT THEN GOTO 750
640 LET FT=1
650 PRINT "TECLEA LAS LETRAS QUE CREAS QUE FALTEN (MAYUSCULAS)"
660 LET A$=INKEY$
670 IF A$="" THEN GOTO 660
680 FOR I=2 TO F-1
690 IF P$(PAL,I)=A$ THEN LET L$(I)=P$(PAL,I);LET LT=1
700 IF L$(I)=" " THEN LET FT=0
710 NEXT I
720 IF LT=0 THEN CON=CON+1:PRINT:PRINT"LO SIENTO,LA LETRA ";A$;" NO ESTA"
730 LET LT=0
740 GOTO 530
750 PRINT "
760 PRINT :PRINT "
770 PRINT ";ENHORABUENA!,LA HAS ACERTADO!";PRINT
780 PRINT "HAS TECLEADO ";CON;" LETRAS ERRONEAS "
790 PRINT "QUIERES JUGAR OTRA VEZ? (S/N)"
800 LET A$=INKEY$
810 IF A$="S" THEN GOTO 240
820 IF A$(">N" THEN GOTO 800
830 REM *****
840 REM ** MODIFICACIONES PARA COMMODORE **
850 REM ** 270 PRINT CHR$(147) **
860 REM ** 530 FOR I=1 TO 8 **
870 REM ** 532 PRINT **
880 REM ** 534 NEXT I:PRINT TAB(10); **
890 REM ** 620 FOR I=1 TO 4 **
900 REM ** 623 PRINT **
910 REM ** 625 NEXT I **
920 REM ** 660 GET A$ **
930 REM ** 740 GOTO 270 **
940 REM ** 800 GET A$ **
950 REM ** MODIFICACIONES PARA SPECTRUM **
960 REM ** 280 GOTO (PAL*10)+280 **
970 REM ** 530 PRINT AT 10,10 **
980 REM ** 620 PRINT AT 15,1 **
990 REM *****

```

El programa comienza con ocho instrucciones DATA, en las que se guardan las palabras que hay que adivinar. Como se observará a la vista del programa, los datos de estas instrucciones no se parecen lo más mínimo a una lista de nombres, sino que son una lista ordenada de números. Estos números precisamente se corresponden con los códigos internos (códigos ASCII) que las letras tienen dentro de nuestro ordenador. Cada número es una letra. La razón de introducir en los DATA los códigos de las letras en vez de las propias letras directamente es obvia: si introdujésemos ahí las palabras que hay que acertar, al teclear el programa en nuestro ordenador tendríamos que teclear estos nombres y cuando posteriormente lo ejecutásemos ya los sabríamos, con lo que el juego perdería su interés. En las instrucciones DATA, al final de los números correspondientes a cada una de las palabras o grupo de palabras consecutivas que hay que acertar se ha colocado un cero, para que el

programa pueda localizar cuando los lea el final de cada uno de ellos. Cuando se trata de un grupo de palabras, el espacio de separación entre éstas se ha señalado aquí con el código 42 (correspondiente al asterisco).

El programa guarda las palabras que lee, después de transformar los códigos en letras mediante la función CHR\$, en la tabla P\$(instrucciones 170-230). Hay veinte palabras o grupos de palabras distintos para acertar y el máximo de letras que pueden tener éstos es de catorce. En aquéllos que tienen menos de catorce letras, se rellenan con blancos los espacios sobrantes. De ahí que, como se ha dicho anteriormente, en los espacios de separación entre las palabras incluidas dentro del grupo que hay que adivinar se ponga un asterisco para diferenciarlos de los del final de dicho grupo.

En la instrucción 240, el programa selecciona un número al azar entre 1 y 20, que será el número de la frase que se escribirá

APRENDER CON EL ORDENADOR

para adivinar las palabras que le falten. Entonces escribe esta frase mediante la instrucción ON (línea 280).

El programa guarda en la lista L\$ la palabra o palabras que hay que adivinar con las letras que tú ya has adivinado poniendo blancos en el resto de las letras que quedan por acertar. Así, inicialmente entre las sentencias 490 y 520 guarda en L\$ la primera y última letra de las que hay que adivinar en sus lugares correspondientes y aquellos caracteres que sean asteriscos, que se corresponden con los espacios de separación entre las palabras y que tampoco hay que acertar.

Entre las instrucciones 530 y 600 se escriben en la pantalla la letra inicial y final, un guión por cada letra que hay que adivinar, así como el contenido de L\$, que será la situación en la que se encuentre la palabra que estamos intentando acertar, excepto los asteriscos, en los que se escribe un espacio en blanco, que es con los que realmente se corresponden.

Cuando introduces una letra (que se guarda en A\$ en la instrucción 660) se comprueba si está, y si es así se guarda en su lugar correspondiente de L\$ (instrucciones 660-710) para luego volver a 530, donde escribe de nuevo L\$ con las letras introducidas.

Significado de las variables de control que utiliza el programa:

ULT = 1 → Indica que en las instrucciones 510-520, en las que se busca la última letra de las que hay que adivinar, aún no se ha encontrado ésta.

ULT = 0 → Lo contrario.

LT = 0 → Indica que la última letra introducida no está en la palabra.

LT = 1 → La última letra introducida sí está en la palabra.

FT = 1 → Indica que ya se ha acertado la palabra.

FT = 0 → La palabra aún no se ha acertado.

F → Número de letras de la palabra o grupo de palabras que se están acertando en cada momento.

Las variables LT y FT se ponen a 0 y 1, respectivamente, cada vez que empezamos una nueva operación (líneas 640 y 730) y si luego se comprueba que lo que indican estos valores no se cumple, se les cambia de valor.

NOTA: El programa, como ya se ha dicho anteriormente, selecciona de manera aleatoria la frase, por lo que puede darse el

caso de que a veces se repitan dos veces casi seguidas, pero, sin embargo, esta técnica, de no seguir un orden preestablecido, aumenta el interés del juego, ya que cuando ejecutes el programa de nuevo en otras ocasiones no sabrás qué frase o frases te van a salir.

PARA LOS MAS JOVENES

■ Manejo del cronómetro

Con este programa que os ofrecemos vais a tener oportunidad de aprender de una manera muy sencilla cómo se maneja un cronómetro.

El programa, al darle la orden de ejecución, escribe en la pantalla inicialmente el cronómetro puesto a cero e indica las teclas que hay que pulsar para conseguir realizar todas las funciones que con él se pueden hacer. Estas funciones se relatan en este cuadro que vemos a continuación:

TECLA	FUNCION
L	Poner en marcha el cronómetro.
P	Parar el cronómetro.
K	Ponerlo a cero.
O	Parar la visualización del cronómetro aunque internamente siga avanzando. Cuando se le vuelva a poner en marcha comenzará por donde fuera internamente.
I	Terminar el programa.

Fig. 10.

Seguidamente vemos también una imagen de cómo está la pantalla durante la ejecución del programa:



Fig. 11.

```

10 REM *****
20 REM **
30 REM **          MANEJO DEL CRONOMETRO          **
40 REM **
50 REM *****
60 CLS
70 GOSUB 1000
80 LOCATE 2,10:PRINT " 0 1 0 1 0"
90 GOSUB 2000
100 FOR I=1 TO 850:NEXT I
110 LET A$=INKEY$
120 IF A$="P" OR A$="O" THEN GOSUB 2000:GOTO 100
130 IF A$="Q" OR A$="O" THEN LET VIS=0
140 IF A$="L" OR A$="I" THEN LET VIS=1
150 IF A$="K" OR A$="K" THEN LET S=0:LET M=0:LET H=0:GOTO 210
160 IF A$="I" OR A$="I" THEN CLS:END
170 LET S=S+1
180 IF S=60 THEN LET S=0:LET M=M+1
190 IF M=60 THEN LET M=0:LET H=H+1
200 IF VIS=0 THEN GOTO 100
210 LOCATE 2,10
220 PRINT H;" ":"M ":" ":"S
230 GOTO 100
1000 LOCATE 15,1
1010 PRINT "TECLAS DE MANEJO DEL CRONOMETRO:"
1020 PRINT "L --> PONER EN MARCHA"
1030 PRINT "P --> PARAR"
1040 PRINT "K --> PONER A CERO"
1050 PRINT "O --> PARAR CON LA CONDICION DE QUE INTERNAMENTE CONTINUE AVANZANDO"
1060 PRINT "I --> TERMINAR EL PROGRAMA"
1070 RETURN
2000 LET B$=INKEY$
2010 IF B$="K" OR B$="K" THEN LET S=0:LET M=0:LET H=0:LOCATE 2,10:PRINT
" 0 1 0 1 0"
2020 IF B$="I" OR B$="I" THEN CLS:END
2030 IF B$<"L" AND B$<"I" THEN GOTO 2000
2040 LET VIS=1
2050 RETURN
3000 REM *****
3010 REM ** MODIFICACIONES PARA COMMODORE **
3020 REM ** Cambiar CLS por PRINT CHR$(147) en 60 y 160 **
3030 REM ** 80 poke 214,2:poke 211,10:PRINT (igual que en listado) **
3040 REM ** 110 GET A$ **
3050 REM ** 210 poke 214,2:poke 211,10 **
3060 REM ** 1000 poke 214,15:poke 211,1 **
3070 REM ** 2000 GET B$ **
3080 REM ** MODIFICACIONES PARA SPECTRUM **
3090 REM ** 80 PRINT AT 2,10 **
3100 REM ** 210 PRINT AT 1,10 **
3110 REM ** 1000 PRINT AT 15,1 **
3120 REM ** MODIFICACIONES PARA ANSTRAD **
3130 REM ** Cambiar el orden de los numeros en los LOCATE de 80,210 y **
3140 REM ** 1000 **
3150 REM **
3160 REM ** NOTA IMPORTANTE! **
3170 REM ** Debido a que la velocidad es distinta en cada ordenador, **
3180 REM ** cada usuario debera ajustar el valor final del FOR de **
3190 REM ** la instrucción 100 para que genere un retardo de un s. **
3200 REM ** cuando para el cronometro **
3210 REM *****

```

■ Comentario del programa

El programa utiliza dos subrutinas que vamos a comenzar explicando para su mejor entendimiento. La primera (que comienza en 1000 y termina en 1070) es la encargada de escribir en la pantalla (a partir de la línea 15 de ésta) la información de las teclas a las que hay

que pulsar para realizar cada una de las funciones del cronómetro. La segunda subrutina (que comienza en 2000 y termina en 2050) es la que se ejecuta siempre que el cronómetro está parado. Obtiene la tecla que se ha pulsado, y si es una K, pone el cronómetro a cero, si es una I, termina el programa, y si no es una L (que es la letra para volver a poner en marcha el cronómetro), vuelve otra vez a la 2000.

APRENDER CON EL ORDENADOR

El programa comienza llamando a la subrutina 1000, a continuación escribe el cronómetro a cero y salta a la subrutina 2000 hasta que se pulse la tecla L para ponerlo en marcha.

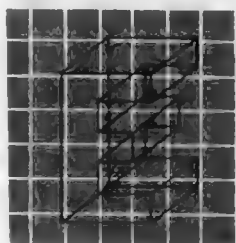
La instrucción 100 realiza el retardo de un segundo para cada ordenador y a continuación se comprueba si se ha pulsado alguna tecla (instrucciones 110-160). Si no se ha hecho, se suma un segundo en las instrucciones 170-190 y se escribe la nueva hora en 210-220, volviéndose de nuevo a la instrucción 100 para retardar un nuevo segundo.

La variable VIS que se utiliza tiene valor cero cuando se quiere que el cronómetro avance internamente, pero no se visualice lo que avance sino que la visualización esté parada hasta que se pulse de nuevo la tecla de arranque. Esta variable se pone, por tanto, a cero cuando se pulsa la tecla O y se vuelve a poner a 1 cuando se pulsa la tecla L.

Las teclas puestas para cada una de las funciones podrían ser cambiadas si el usuario desea tener otras, sustituyéndolas en las instrucciones IF en las que aparece cada una de ellas.

PEQUEÑA HISTORIA DE LA INFORMATICA

■ Primeros ordenadores americanos



N el diseño de calculadores, Von Neumann había dado un enorme paso hacia adelante. A partir de ese momento, casi la totalidad de los ordenadores que irían apareciendo se basarían en su estructura. Sin embargo, a pesar de haberse hecho famoso por el diseño del EDVAC, no encontraba a nadie que pudiera hacerse cargo de dirigir la fabricación de la máquina. Sus antiguos compañeros de aventuras, enfrentados con él, posiblemente por tener dos visiones muy diferentes de la vida, habían decidido abandonar la Moore School y aventurarse en la creación de una empresa. Mauchly se seguía considerando uno de los "padres" del EDVAC, y así lo propagaba en artículos (que decían casi las mismas cosas que los de Von Neumann). Según él, el principio más original de la máquina era su memoria, grande, en la que se guardaban instrucciones y datos. Efectivamente, ese fue el mejor acierto que tuvo el equipo, y muchas máquinas posteriores seguirán la misma trayectoria.

Eckert y Mauchly se pusieron inmediatamente a trabajar en otra nueva máquina, que sería más rápida (unas cuatro veces) que el EDSAC inglés y bastante más pequeña que el ENIAC. Disponía de muchos menos tubos que éste, y sumaba mucho más lentamente, pero las multiplicaciones eran muchísimo más rápidas (tres veces más rápidas). Su memoria era de línea de retardo de mercurio, con una capacidad enorme ¡512 palabras de 31 bits! La máquina tenía otro aspecto nuevo: estaba formada por dos procesadores idénticos, que realizaban las operaciones en paralelo, confrontando los resultados, que debían ser idénticos y deteniéndose en caso contrario.

La máquina comenzó a funcionar en abril de 1949, y durante su presentación funcionó bien, sin producirse averías. Sin embargo, su resultado no fue bueno, y decían las lenguas afiladas que "sólo funcionaba cuando hacía buen tiempo, negándose a ello los días lluviosos". Es una pena que no pudiera mantener la magnífica cota de averías obtenida durante su presentación ¡cuarenta y cuatro horas de trabajo sin averías! La compañía que contrató la máquina, la Northrop Aircraft Company, puede enorgullecerse de haber sido una de las primeras (si no la primera) compañía americana que adquirió un ordenador.

Poco después del BINAC, en el Laboratorio de Servomecanismos del Massachusetts Institute of Technology (MIT) fue creado otro nuevo calculador u ordenador: el Whirlwind (su significado es "torbellino"). El proyecto era, en principio, para la Marina Americana, pero más tarde pasó a las Fuerzas Aéreas. La utilidad que pensaba darse a la máquina era principalmente el control y gestión de uno de los primeros simuladores de vuelo, elemento que sería fundamental en el entrenamiento de pilotos, pero más adelante se decidió utilizarla para otras múltiples tareas, de todo tipo, especialmente las que requerían trabajar en tiempo real, donde la máquina se revelaba bastante eficaz.

(Como es la primera vez que aparece el término "tiempo real", indiquemos que un ordenador que trabaje "en tiempo real" debe ser capaz de procesar los datos y enviar los resultados al objeto, en un tiempo lo suficientemente pequeño para que éste pueda modificar su comportamiento con la información que recibe del ordenador.)

El proyecto se comenzó en octubre de 1944, pero la construcción tardó mucho más en arrancar, emprendiéndose en 1946. Fue terminada en 1951.

PEQUEÑA HISTORIA DE LA INFORMATICA

El Whirlwind fue una máquina muy nueva en muchos aspectos. Disponía de muchos elementos absolutamente desconocidos, hasta el momento, algunos de ellos muy efectivos, como un terminal gráfico de presentación de resultados, y un lápiz electrónico para el intercambio de información entre el operador y la máquina. Su memoria era de toros magnéticos. En cuanto al software, disponía de unas rutinas de diagnóstico que permitían detectar los errores mucho más fácilmente, y de otras ayudas para, por ejemplo, unir distintas partes de un programa, etc. El lenguaje de programación que se utilizaba no era lenguaje máquina, resultando la programación mucho más simple y agradable. Pero sus ventajas no terminan aquí; la máquina podía controlar otras máquinas-herramienta, reconocer formas o patrones, y comunicarse con otro ordenador distante, enviando los datos por línea telefónica. Era una verdadera maravilla.

Su diseñador fue Jay Forrester (fundador del Digital Computer Laboratory). A él se debe también el primer prototipo de memoria de toros de ferrita. La máquina era la más rápida de su época, y, por tanto, muy adecuada para controlar los incipientes simuladores de vuelo, comprobando todos los instrumentos que manejaba el piloto, y devolviendo las respuestas casi simultáneamente. Además, también resultaba muy útil para probar las características de los nuevos modelos aerodinámicos que iban apareciendo. (En los años cincuenta cada nuevo tipo de avión requería un nuevo simulador de vuelo.)

Los Whirlwind pasaron a formar parte del sistema de defensa aérea de los Estados Unidos, SAGE (Semi Automatic Ground Environment), realizando simulaciones y otras muchas tareas muy útiles para las Fuerzas Aéreas. Existían varias versiones, entre las que debemos destacar la AN/FSQ 7 y 8, creada especialmente para el sistema SAGE, que estaba dispuesta en una amplia sala, rodeándola como una biblioteca. El control y manejo de la máquina se efectuaba desde otra sala de control. La máquina era muy grande (5.000 tubos y más de 11.000 diodos). Utilizaba palabras de 16 bits por primera vez, y realizaba multiplicaciones en varios microsegundos. Su memoria era de 32 CRT.

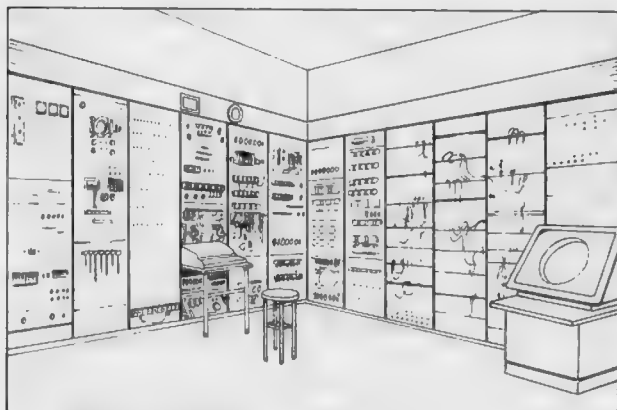
¿Cuál era, pues, el fallo de esta increíble máquina? Casi el mismo que tenían todas sus homólogas. Sus averías, y de ahí, su poca fiabilidad. Es bien sabido que el material que

¿Sabía usted que...

Un ordenador típico de los años cincuenta estaba compuesto por los siguientes elementos: un armario lleno de válvulas termoiónicas, resistencias, condensadores, cables, conectores, etc. Otro armario más pequeño para la energía necesaria, una unidad de entrada (lectora de cinta perforada o de papel), unidad de salida (de cinta o de tarjeta perforada) y una consola para el operador. En el armario principal estaba localizada la unidad central de proceso (que es la que dispone de los circuitos electrónicos que realizarán las operaciones de cálculo, de control o de memoria). Esta última (la memoria) solía estar en el armario principal, pero podía encontrarse separada. Frecuentemente se trataba de una memoria de línea de retardo de mercurio, muy sensible a los cambios de temperatura, por lo que si era de este tipo, debía colocarse separada, en otro armario. Otras memorias de tambor magnético, algo más modernas, utilizadas con frecuencia como memoria secundaria, no tenían estos inconvenientes, y podían ser introducidas en el armario principal.

utiliza el Ejército suele ser mucho más caro que otro material parecido comprado en tiendas. La razón no es que las industrias timen al Ejército. El material que utiliza éste es en muchos casos idéntico al que puede adquirirse en una tienda, pero la diferencia fundamental está en las pruebas que ha sufrido. Si un transistor, una tela, o una fibra se someten a pruebas en ciertas condiciones ambientales normales, o forzando algo las condiciones, en el caso de que fueran para el Ejército se someterían a otras muy distintas, forzando muchísimo más las condiciones (temperatura, humedad y otros agentes). Sólo el material que supere las pruebas será aceptado, rechazándose el resto. Esta selección encarece mucho los precios, pero alcanza cotas de fiabilidad muy superiores a las que se ofrecen en el mercado normal. La fiabilidad de un dispositivo es, pues, una cualidad fundamental para el Ejército. Y el Whirlwind ("torbellino") disponía de demasiados tubos de vacío para presentar una fiabilidad mínimamente aceptable. La máquina más veloz de su tiempo pasaba horas reparándose, con lo que su utilidad real disminuía enormemente.

¿Qué ocurría mientras tanto con Von Neumann? No, no pasó al olvido; seleccionó la



El ordenador Whirlwind.

Universidad de Princeton para trabajar en su próximo proyecto, el IAS. Von Neumann rompió así una tradición de esta Universidad, muy elitista, que aceptaba difícilmente las implicaciones (contratos, fondos externos, etc.) que traía consigo la fabricación de un calculador universal. Von Neumann se llevó consigo a la Universidad a Hermann Goldstine, con el que seguía teniendo magníficas relaciones. (Goldstine, Von Neumann y el matemático Burks estando en la Moore School, habían escrito un pequeño artículo en el que se basaría el nuevo trabajo.)

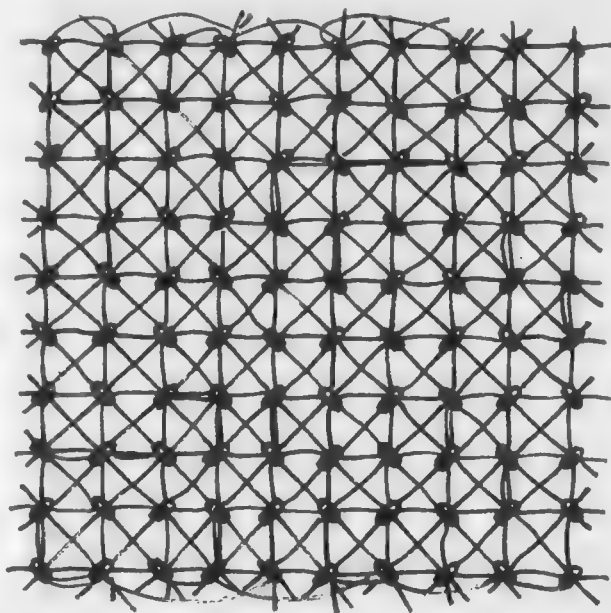
Von Neumann no paró, convenció para que le financiaran el producto, consiguió que le permitieran trabajar en el Institute of Advanced Study, de la Universidad de Princeton, y contrató a un protegido de Wiener, Julián Bigelow, como director del proyecto. (Wiener era otro investigador de talla, entre otros campos en el de las Matemáticas). Sin embargo, no está claro si esta última decisión de contratar a Bigelow fue un acierto, ya que Bigelow, aunque era ingeniero, era muy teórico y poco práctico. (Bigelow estaba muy interesado, como Von Neumann, en comparar las características de las nuevas máquinas con las funciones del cerebro humano. En 1942 escribió un artículo, junto con Wiener y Rosenbluth, que constituye la base de la cibernética.)

La memoria de la nueva máquina de Von Neumann estaba construida a base de tubos electrostáticos de mercurio, y era algo más rápida que la de líneas de retardo de mercurio. Las palabras se leían en paralelo, no secuencialmente, lo que resultaba una ventaja adicional. Sin embargo, con el tiempo esa tecnología no tuvo mucho éxito, terminando por abandonarse.

No podemos seguir hablando de los distintos ordenadores que fueron apareciendo durante los primeros años de la década de los cincuenta, sin hablar de las memorias, principal preocupación de diseñadores y fabricantes de ordenadores.

Desde 1950 hasta 1955, aproximadamente, los constructores de los nuevos ordenadores estaban preocupados principalmente en las memorias. Efectivamente, el primero de ellos era Von Neumann, que basaba el futuro de las nuevas máquinas en obtener una mayor fiabilidad con los tubos de vacío, que iban enriqueciéndose con los años) y mejores memorias, más amplias y con un acceso más rápido. Los tubos de vacío necesitaban tensiones menores, dato importante, porque disipaban menos y duraban bastante más (además de ocupar mucho menos). En cuanto a la memoria, se adoptaron distintas soluciones, y sólo a mediados de la década de los cincuenta se pudo disponer de memorias aceptables, ocupándose los investigadores en este campo en otros órganos de la máquina.

Veamos muy por encima qué es una memoria. La memoria de un ordenador es un dispositivo que permite el almacenamiento de los datos, y su recuperación posterior. El tamaño de la memoria se denomina su "capacidad" y se mide indicando el volumen de información que puede almacenar. La información se organiza en grupos de bits, palabras o caracteres que ocupan distintas posiciones o direcciones de memoria. A cada posición de la



Núcleos de ferrita

PEQUEÑA HISTORIA DE LA INFORMATICA

memoria se puede acceder a través de su dirección. El tiempo de acceso a la memoria es aquel necesario para grabar o recuperar un dato de una posición de la memoria. El ciclo de memoria, normalmente, es dos veces el tiempo de acceso a la memoria. Las distintas memorias de un ordenador están organizadas jerárquicamente. Existe, pues, una memoria principal, cuyo tiempo de acceso es muy pequeño. Son las que se comunican directamente con la unidad central de proceso. Otras memorias de tamaño muy superior son las memorias secundarias, que no se comunican directamente con la unidad central de proceso ni se utilizan, por tanto, directamente en los cálculos, pero sirven para almacenar mucha más información, que puede pasar después a la memoria principal cuando vaya a ser utilizada. Las tecnologías que se utilizaron en los primeros años cincuenta para la fabricación de memorias fueron tubos de vacío, líneas de retardo, otros tipos de tubos electrostáticos, tambores magnéticos y toros de ferrita.

Otra nueva división de las memorias puede ser en dinámicas y estáticas. En las memorias estáticas, la información permanece (biestable, etc.). En las memorias dinámicas, la

información circula por el dispositivo, aunque sólo puede recuperarse en un cierto punto. A este último tipo pertenecen las memorias de "línea de retardo", en las que la información (señal) va circulando, y es reforzada cada cierto tiempo, para evitar que se vaya atenuando hasta desaparecer prácticamente. Los tambores magnéticos han sido utilizados tanto para memorias principales como para memorias secundarias. Su superficie está recubierta por una sustancia que puede magnetizarse de forma discreta. Las memorias de tubos electrostáticos no eran muy fiables porque eran especialmente sensibles a perturbaciones electromagnéticas, produciéndose demasiados errores. Además, su vida media era muy corta. Las memorias de toros de ferrita fueron la mejor solución en los años cincuenta al problema de obtener memorias fiables, y con tiempos de acceso pequeños. Su utilización ha durado muchos años. De todas las memorias mencionadas, hablaremos mucho más ampliamente en otro momento. Baste este resumen cortísimo para que el lector pueda ir siguiendo las mejoras surgidas en este dispositivo en las distintas máquinas que se fueron creando.

■ Ordenadores vectoriales



AMOS a hablar de los ordenadores vectoriales que son un tipo especial de ordenadores no secuenciales. Los ordenadores convencionales (arquitectura de Von Neumann) trabajan con los datos de un secuencial (escalar), mientras que en los ordenadores vectoriales se manejan los datos en bloque (vectores).

Los ordenadores vectoriales utilizan ciertas formas de paralelismo (a nivel hardware) en el procesamiento de los datos. Naturalmente, no se podrá sacar partido de sus ventajas, si las aplicaciones no se adecúan específicamente a la arquitectura especial de estas máquinas.

Veamos, en primer lugar, una definición de vector. Desde el punto de vista de los ordenadores vectoriales, un vector es un conjunto de información que se procesa colectivamente, es decir, a la que se aplican colectivamente las operaciones de que se trate. Dichas operaciones se ejecutan mediante estructuras pipeline.

Hemos hablado ya de ordenadores paralelo, aunque hay que considerar muchos niveles de paralelismo, desde programas, modos de explotación, etc. En general, el concepto de paralelo es opuesto al de escalar, ya que, utilizando la tecnología adecuada, el ordenador vectorial ejecuta "simultáneamente" varias tareas que tradicionalmente realizaba de forma secuencial el ordenador escalar. Como resultado, se puede procesar mucha más información, y al mismo tiempo se obtienen velocidades muy superiores (recordemos que muchos trabajos y proyectos no podían ni siquiera estudiarse, ya que con los métodos tradicionales, los resultados se obtendrían en un momento en el que ya no tendrían interés alguno, por ejemplo, investigación meteorológica,

o en general astronómica, sistema de visión artificial, etc.).

La ejecución de una operación vectorial sobre un conjunto de vectores implica necesariamente utilizar elementos específicos. Los componentes funcionales de estos elementos se encuentran en la unidad vectorial.

Sin embargo, el ordenador vectorial necesita ciertas funciones y no siempre están incluidas todas ellas en la unidad vectorial. Un punto que es muy importante aclarar es que en una determinada aplicación existirán partes susceptibles de ser tratadas vectorialmente, y otras cuyo tratamiento óptimo no sea de este tipo. Así, pues, como todas las aplicaciones no son adecuadas exclusivamente para un tratamiento vectorial, deberán ser ejecutadas en parte en un ordenador vectorial, y (la parte correspondiente) en un ordenador escalar. Por esta razón, todos los ordenadores vectoriales disponen siempre de una unidad escalar.

Como los ordenadores vectoriales se han diseñado específicamente para obtener las mejores características de este tipo, específicamente, no son los mejores para realizar gran número de tareas generales (gestión de una base de datos, etc.), y, por tanto, estas tareas las realiza un ordenador frontal (escalar) unido al ordenador vectorial canal a canal. De este modo se dividen las tareas, de forma que las que no sean directamente de carácter vectorial, pasen a ser tratadas por el ordenador escalar (sistema frontal).

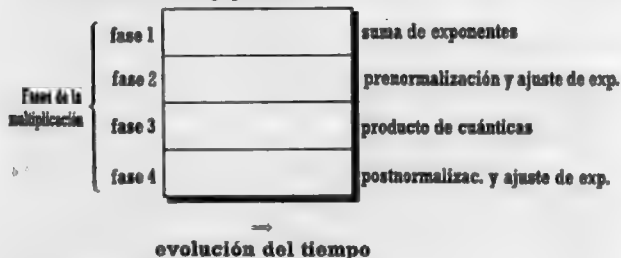
Hemos indicado unas líneas antes, que cualquier ordenador muy especializado sólo será útil si las aplicaciones concebidas para ser ejecutadas en él se adaptan perfectamente a su estructura. Así, se podrá sacar un mayor partido de las cualidades vectoriales del ordenador cuanto más adaptada a su estructura vectorial esté la aplicación.

Si consideramos dos vectores (tales que el número de elementos que lo forman es igual), su multiplicación se realizará utilizando la conocida técnica de la coma flotante.

A) Vectores a multiplicar

$x_1, y_1, z_1, v_1, u_1, t_1 \dots \dots \dots w_1$
$x_2, y_2, z_2, v_2, u_2, t_2 \dots \dots \dots w_2$

B) Estructura del pipeline



C) Realización del producto

x_1-x_2	y_1-y_2	z_1-z_2
	x_1-x_2	y_1-y_2
		x_1-x_2

1.º ciclo

2.º ciclo

3.º ciclo

v_1-v_2	u_1-u_2	t_1-t_2
z_1-z_2	v_1-v_2	t_1-t_2
y_1-y_2	z_1-z_2	v_1-v_2
x_1-x_2	y_1-y_2	z_1-z_2

4.º ciclo

5.º ciclo

6.º ciclo

(concluye el producto de x_1-x_2)

(concluye el producto de y_1-y_2)

(concluye el producto de z_1-z_2)

(La coma flotante es un sistema muy útil de representar los números para su posterior manipulación. El número se expresa mediante dos cantidades, una mantisa y un exponente, al que debe elevarse la base de la representación —usualmente 10— para que después, multiplicando esta potencia por la mantisa, se pueda obtener el número real del que partimos. El sistema es muy cómodo. Las cantidades se perciben con mucha mayor claridad, ya que inmediatamente se aprecia su orden de magnitud. Además, las operaciones resultan mucho más sencillas. Para que un ordenador acepte expresar los números con coma flotante, debe disponer de unas instrucciones específicas, para que el usuario pueda expresar las cantidades que maneja en sus programas, con coma flotante. La precisión del número expresado en coma flotante viene dada por el número de cifras de su mantisa.)

Los ordenadores vectoriales están estructurados con doble precisión. Este último término indica que los órganos internos que componen la máquina (registros, circuitería,

etcétera) permiten la transferencia de bloques de información de sesenta y cuatro bits (ocho octetos). Si la información está especificada con simple precisión, la máquina deberá primero convertir la información a doble precisión, antes de pasar a procesarla.

Las etapas de una multiplicación normal en coma flotante son cuatro: suma de los exponentes, prenormalización y primer ajuste de los exponentes, multiplicación de las mantisas y posnormalización y ajuste consiguiente de los exponentes.

Así, en un ordenador escalar la multiplicación se realiza pasando sucesivamente por los cuatro pasos que acabamos de mencionar. (En resumen, la multiplicación utilizará cuatro ciclos de máquina, y será necesario esperar al quinto para que pueda comenzar a efectuarse la multiplicación del y_1 por y_2 , y así sucesivamente). (*)

En los ordenadores escalares, la multiplicación de dos números escritos en coma flotante se ha realizado de modo que el producto de cada pareja de cifras se realiza secuencialmente (por ejemplo, $x_1.x_2$; $y_1.y_2$; $z_1.z_2$, etcétera.)

Veamos cómo se realiza la multiplicación con un sistema vectorial. En la figura podemos ver cuatro líneas que se corresponden con las cuatro fases de la multiplicación en coma flotante. En primer lugar, tenemos la fase de suma de los exponentes. En segundo lugar, la prenormalización y el ajuste de los exponentes, etc.

Si nos fijamos en las posiciones verticales, puede seguirse fácilmente el paso de los datos por el pipeline (tubería), con el encadenamiento de las fases de cada operación.

Veamos, por ejemplo, al comienzo del primer ciclo, en el que los datos x_1 y x_2 entran en el pipeline. Al final de dicho ciclo se habrá realizado la primera fase de la multiplicación, es decir, la adición de los exponentes, y el resultado pasa a la parte de pipeline que realiza la segunda fase de la operación, encargándose del ajuste y prenormalización. Pero, en ese momento, "queda libre" la primera parte del pipeline, y puede entrar el segundo par de valores, y_1, y_2 . A partir de entonces el proceso se realiza simultáneamente. Durante el segundo ciclo, el grupo de valores x realiza su

(*) La normalización es la operación consistente en poner la primera cifra significativa delante de la coma decimal, y el resto de las cifras detrás.

segunda fase de multiplicación, mientras que el par y realiza la primera. Una vez finalizado el proceso, el conjunto pasa a la fase siguiente (la pareja y_2 - y_2 a la segunda, y la pareja x_1 - x_2 a la tercera), y así sucesivamente. Durante el cuarto ciclo, las x 's pasan a la última fase de la operación (posnormalización y ajuste del exponente), y al final del ciclo se habrán completado las cuatro fases de la multiplicación y se habrá obtenido el resultado. Pero si en un ordenador escalar la multiplicación duraba cuatro ciclos, ¿cuál es la ventaja fundamental de los ordenadores paralelo?

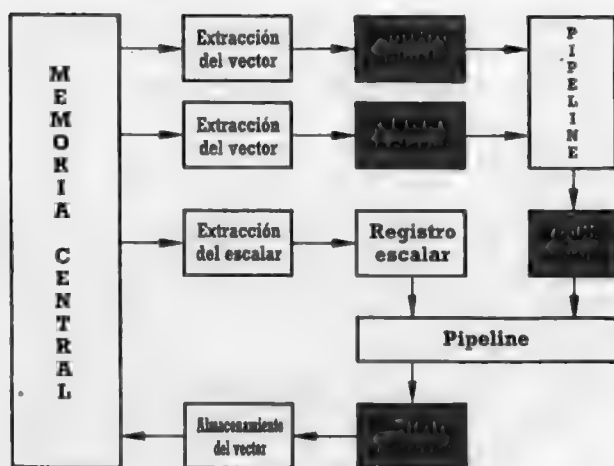
La ventaja aparece a partir del quinto ciclo, y con tal de que existan datos para alimentar la entrada del pipeline. El sistema pipeline supone un paralelismo, pero no a nivel de operaciones, sino de las fases de dichas operaciones. No parece muy claro que estos sistemas resulten más rápidos y apropiados que los escalares para el tratamiento de datos. Sin embargo, recordemos que son ordenadores *vectoriales*, es decir, trabajan con vectores que pueden tener dos o más componentes. Si cada vector tiene n componentes, en un equipo tradicional escalar (arquitectura de Von Neumann), y puesto que la multiplicación tarda cuatro ciclos, el proceso completo tardará $4n$ ciclos. En los ordenadores que disponen de estructuras pipeline, a partir del tercer ciclo (en que ya comienzan a introducirse nuevos valores en el pipeline se va obteniendo un resultado por ciclo, y, por tanto, la duración será $3 + n$ ciclos. Queda, pues, claro que las estructuras pipeline "pierden" un pequeño número de ciclos, al comienzo de cada operación, independientemente de la longitud de los vectores que se estén procesando. A estos ciclos sin resultado se les designa con el nombre inglés "start up", y su número es igual al número de fases de la operación a realizar, menos uno. (De la explicación anterior puede verse que este tiempo improductivo —como se ve— no depende en absoluto del número de componentes de cada vector, y al contrario, cuanto más corto sea éste, más repercusión tendrán sobre él los "tiempos muertos" del arranque de la estructura pipeline.)

Como hemos visto, para que una estructura pipeline funcione adecuadamente, los datos deben introducirse en la estructura al ritmo adecuado, sin que se produzcan pérdidas de tiempo (la estructura pipeline debe estar alimentada permanentemente, ya que esta particularidad es la que la hace rentable, pasados los primeros ciclos de arranque. Nor-

malmente, los datos que pasan al pipeline provienen de la memoria central del sistema, por lo que ésta deberá efectuar la transferencia a la velocidad pedida. Esta se mide en megapalabras por segundo (o megaoctetes por segundo) y se denomina ancho de banda.

El problema siguiente aparece cuando las unidades centrales son más rápidas que las memorias centrales, caso, por lo demás, muy común, puesto que la mayor parte de los ordenadores vectoriales han sido diseñados para procesar gran cantidad de información, y para aumentar la velocidad de proceso en aquellos casos en los que, o existe efectivamente mucha información a procesar, o la velocidad de cómputo es un objetivo primordial a conseguir. Naturalmente, los fabricantes de ordenadores intentan equilibrar en lo posible estas diferencias de velocidad entre la memoria y la unidad central, utilizando algunos recursos específicos de estos ordenadores. Así, aparecen muchos tipos de memorias intermedias, entrelazadas, locales (de registro), etc. Estas últimas son características de los ordenadores vectoriales, e indican con bastante exactitud las características de la máquina de que se trate. En realidad, estas memorias centrales (y construidas con otra tecnología), podrían utilizarse como memoria central, pero son bastante más caras, por lo que suelen formar parte de ésta, pero sus dimensiones son mucho menores. Están diseñadas para que sus ciclos de base se acomoden a los de la unidad central.

Veamos a continuación el uso de estos registros de memoria. Cada vector se almacena en un registro de esta memoria especial, y el aprovechamiento consiste en enviar el resultado de la operación realizada con los vectores (que es otro vector) a otro registro, en lugar de hacerlo a la memoria central, de forma que si la operación posterior es, por ejemplo, una suma o multiplicación por un escalar, éste se encuentre en otro registro escalar, y una vez efectuada la operación, el resultado se envíe a otro registro. (Es un método para acceder a la memoria central lo menos posible). A este nuevo sistema de operar propio de los ordenadores vectoriales se le denomina "encadenamiento de registro". Con él se evita el envío de los resultados intermedios a la memoria central, evitando muchas pérdidas de tiempo en estas transferencias de datos. Es claro que cuanto menos se necesite recurrir a la memoria central, menos retenciones existirán, y más se aprovechará la estructura pipeline.



Para que se produzca un encadenamiento en determinadas operaciones, es necesario que el hardware de la máquina lo permita. Además, en la operación también están implicados compilador e incluso en ciertos casos se podrá recurrir al ensamblador.

Los registros temporales tienen una importancia capital en el encadenamiento, pero éste necesita además imprescindiblemente que los resultados intermedios pasen a la estructura pipeline nada más ser producidos, para seguir el curso de las operaciones. En los casos en los que la estructura pipeline aprovecha el encadenamiento, los tiempos de ejecución (en ciclos) para una multiplicación y adición, serán: el tiempo debido al arranque del pipeline, más el debido al arranque del pipeline suma, más el correspondiente a la longitud de los vectores (número de pares de valores a multiplicar), es decir, se realizan dos operaciones por ciclo. Si la operación se lleva a cabo de forma secuencial, habrá que considerar, en esta suma de tiempos el arranque del pipeline multiplicación, el correspondiente a la longitud de los vectores (número de pares a multiplicar), el arranque del pipeline suma, y de nuevo habrá que considerar la longitud de los vectores. Así pues, tendremos $am + as + lv$ como tiempo de ejecución total, medido en ciclos para la ejecución completa de las dos operaciones. Si se hubiera realizado de forma secuencial, el tiempo a considerar sería la suma del tiempo de arranque de la pipeline multiplicación, más el correspondiente a la longitud de los vectores, más el de arranque del pipeline suma, más otra vez el correspondiente a la longitud de los vectores. Será, por tanto, $am + lv + as + lv$ más largo que el anterior, es decir, con el encadenamiento se efectúan dos operaciones por cada ciclo.

Veamos ahora otro término que se utiliza con mucha frecuencia para comparar las características de varias máquinas (aunque no es un término que mida con exactitud la potencia de éstas). El "peak rate" mide el número de veces que se ejecuta en un segundo una secuencia multiplicación-adición. Sin embargo, hemos indicado varias veces que se pueden obtener resultados muy diferentes con un mismo ordenador vectorial, y distintas aplicaciones. Los resultados dependen mucho del aprovechamiento de las características que están a disposición del programador o del analista. De ellos depende en gran medida la velocidad de cómputo del ordenador e incluso otras características. Para obtener un mayor aprovechamiento, es necesario que la aplicación se acople perfectamente a todos los recursos de que dispone. Ya sabemos que el encadenamiento no se efectúa automáticamente en todas las operaciones.

Como la razón de ser de los ordenadores vectoriales frente a los escalares es poder absorber tareas que impliquen gran cantidad de información (algunas de estas tareas resultan inabordables por los ordenadores escalares), la velocidad de cómputo es un elemento muy importante. (Piénsese en un ordenador de un instituto meteorológico, que deba dar predicciones de tiempo, manejando mucha información, y cuyos resultados aparezcan al día siguiente del día estudiado, o en un ordenador de un equipo de visión artificial, que debe procesar la información recibida de forma "casi instantánea" para que el resultado sea una imagen mínimamente aceptable. Para acrecentar dicha velocidad, se puede aplicar de nuevo el concepto de paralelismo con las estructuras pipeline. A este nuevo sistema se le designa con el término inglés "overlap". Véase la figura. El "overlap" puede también aplicarse a los tiempos de arranque de las estructuras pipeline, y consiste esencialmente en colocar en paralelo dos (o más) estructuras pipeline que ejecuten distintas operaciones. En algunos ordenadores se repite totalmente el conjunto de pipelines varias veces. Sus ventajas y limitaciones son las propias de los sistemas paralelo, y los resultados dependen mucho del programador o analista que puede adaptar perfectamente la aplicación a las estructuras de la máquina, o desaprovecharla en gran parte. La codificación (código fuente o resultante) debe, pues, tener en cuenta el paralelismo para obtener las velocidades potenciales de la máquina.

TERMINOLOGIA

Ciclo. Conjunto de tareas secuenciales elementales que se repiten periódicamente para la realización de un proceso global.

Ciclo de base. Dícese del conjunto de tareas elementales que realiza el ordenador en cada período de tiempo. Intervalo mínimo de realización de una instrucción elemental.

Coma flotante. Sistema de representación de las cantidades numéricas mediante dos números: mantisa (conjunto de cifras representativas de la cantidad numérica, con la precisión deseada) y exponente (número al que hay que elevar la base de la representación, normalmente diez, y que da el orden de magnitud de la cantidad representada).

Encadenamiento de registros. Ligazón que se establece entre varios registros de tal modo que se pueda producir el paso automático de los datos de uno a otro registro, mejorando, en consecuencia, la velocidad de proceso.

Normalización. En la multiplicación de dos números en coma flotante, se llama así a la operación consistente en poner la mantisa con una sola cifra delante de la coma decimal, y las restantes cifras detrás de dicha coma decimal.

Ordenador escalar. Se llama así a los ordenadores convencionales, arquitectura de Von Neumann, en contraposición a los ordenadores de proceso paralelo o vectorial.

Ordenador frontal. Dícese del primer ordenador que aparece a la "entrada" de una estructura compleja de procesadores. En una red de datos suele ser el ordenador que controla los protocolos de comunicación, recibe los datos de la red exterior y los transfiere al (o a los) ordenadores de proceso. En un sistema de procesamiento paralelo, suele ser un ordenador de tipo escalar, que se comunica con el exterior y gestiona periféricos.

Ordenador vectorial. Ordenador que opera con vectores, procesando varios elementos del vector simultáneamente e introduciendo por tanto alguna forma de paralelismo en el proceso de los datos.

Overlap. Solapamiento de datos o programas para el aprovechamiento del espacio de memoria disponible. Cuando existe un solapamiento en la realización de operaciones aritméticas, el proceso recibe el nombre de conducción paralela, o pipelining.

Paralelismo. Sistema de proceso simultáneo de diferentes bloques de código ejecutable o de distintas fases, dentro de una instrucción compleja, realizadas de este modo para aumentar la velocidad de proceso del ordenador. Existen diversas formas de paralelismo, dependiendo de la arquitectura del equipo que la realiza según, se solapen diversas instrucciones o los procesos de acceso a memoria con los de proceso de datos, etc.

Peak rate. Valor que indica el número de veces por segundo que se efectúa una multiplicación-adición en un ordenador vectorial. Es un sistema algo burdo de medir la potencia de la máquina que se está analizando, frente a las demás. También se denomina "megaflops maximum".

Pipeline, estructura. Estructura compleja, propia de los ordenadores vectoriales o de proceso paralelo, mediante la que se encaminan o conducen (pipelining) inmediatamente los datos de salida de una fase del proceso al punto en el que se va a realizar la siguiente fase del proceso.

Registros. Conjunto de dispositivos que se utilizan para el almacenamiento de información, dentro de un ordenador de gran velocidad de acceso. Frecuentemente tienen el tamaño de una palabra de ordenador, un byte, etc. La configuración almacenada en un registro puede representar una instrucción, un número binario o un carácter alfanumérico.

Vector. Conjunto de información cuyo proceso se efectúa colectivamente, es decir, al que se aplican de forma colectiva las operaciones de que se trate. Para el proceso de vectores se utilizan unas estructuras especiales llamadas "de conducción", o pipeline.

Object code. Término inglés para designar al código objeto. Salida de compilador o ensamblador que en sí misma es un código máquina ejecutable, o al menos adecuada para producir código máquina ejecutable.

Object language. Término inglés que indica un lenguaje al que se traducen las instrucciones desde un lenguaje de alto nivel.

Object program. Código objeto estructurado para constituir un programa completo.

OCR (Optical Character Recognition). Reconocimiento óptico de caracteres. Técnica que utiliza un dispositivo óptico para reconocer caracteres escritos. Se alude en ocasiones, con este término, al proceso (software) de identificación de los distintos caracteres. Se habla también de caracteres OCR o documentos OCR designando a caracteres (o documentos que contienen caracteres) especiales capaces de ser reconocidos por un dispositivo físico adecuado.

Octal. Sistema de representación numérica en base ocho. Puede referirse a un conjunto de ocho posibilidades, de la que es preciso seleccionar alguna.

Octeto. 8 bits. Es equivalente a un byte. Porción de una palabra de datos de un ordenador.

Odd. Término inglés que significa impar.

OEM (Original Equipment Manufacturer). En muchas ocasiones, los fabricantes de equipos industriales compran sistemas (ordenadores electrónicos), partes o componentes, para incorporarlos a sus equipos, o incluso revenderlos modificando alguna de sus características.

Off line. Término que se aplica a dispositivos o ficheros no conectados directamente al ordenador, sobre los que, por tanto, la unidad central de proceso no tiene control.

On line. Término inglés para designar que un dispositivo, archivo o, en general, un recurso se encuentran bajo el control del programa. En contraposición a off line. Los dispositivos o en general cualquier recurso «on line» están disponibles inmediatamente para ser utilizados. Un ejemplo puede ser un fichero del disco que usted está utilizando en este momento.

Open. Término inglés que significa abrir o abierto. En informática se refiere a las instrucciones que sirven para «conectar de forma lógica» programas y periféricos, abriendo un canal.

Operador. Persona encargada de realizar tareas de carga de programas, cambio de discos, etc., en un centro de proceso de datos.

Símbolo que define la acción a desarrollar por los operandos. En matemáticas, la suma, resta, etc., son operadores. En informática se utilizan tres tipos fundamentales de operadores, aritméticos (suma, resta, producto, división y elevación a potencias, etcétera), lógicos (AND, OR, NOT, XOR, NAND, NOR...) y operadores de relación (menor que <, menor o igual que <=, mayor que >, mayor o igual que >=, distinto o diferente de <>...).

También el término operador puede designar a una instrucción de máquina.

Operador diádico. Operador que actúa sobre dos operandos únicos.

Operando. Argumento sobre el que actúa un operador, es decir, en una multiplicación, multiplicando y multiplicador. También se indica con este nombre (en lenguaje ensamblador o lenguaje máquina) a los valores que hay que combinar con los datos del registro base para obtener las direcciones de la memoria donde se encuentran los operandos sobre los que debe ejecutarse la instrucción. En este caso, el direccionamiento sería indirecto o indexado.

